FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA

Francisco José Barreto Nunes

PASS
PROCESSO DE APOIO
À SEGURANÇA DE SOFTWARE

Fortaleza
2007

FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA

Francisco José Barreto Nunes

PASS
PROCESSO DE APOIO
À SEGURANÇA DE SOFTWARE

Dissertação submetida ao Corpo Docente do Mestrado em Informática Aplicada da Universidade de Fortaleza como parte dos requisitos necessários para a obtenção do grau de Mestre em Informática Aplicada.

Orientador: Prof. Dr. Arnaldo Dias Belchior (In Memorium)

Fortaleza
2007

Francisco José Barreto Nunes

PASS
PROCESSO DE APOIO
À SEGURANÇA DE SOFTWARE

Data de Aprovação:

Aprovada por:

_____

Prof. Pedro Porfirio Muniz Farias, D.Sc. - UNIFOR

_____

Prof. Alexandre Marcos Lins de Vasconcelos, D.Phil. - UFPE

_____

Prof[a]. Maria Elizabeth Sucupira Furtado, Docteur. - UNIFOR)

N972p Nunes, Francisco José Barreto.
PASS – Processo de Apoio à Segurança de Software / Francisco José Barreto Nunes – 2007.
203 f.

Cópia de computador.
Dissertação (mestrado) – Universidade de Fortaleza, 2007.
"Orientação : Prof. Dr. Arnaldo Dias Belchior."

1. Software. 2. Segurança de computador. I. Título.

CDU 681.3.06

DEDICATÓRIA

Dedico este trabalho à memória de meu amado pai Francisco José Sales Nunes e à memória de meu amado avô José Adonias Moreira que sempre acreditaram na perseverança e me apoiaram e deram provas de carinho, amor, respeito e dedicação.

Dedico também este trabalho à memória do meu estimado amigo e orientador Arnaldo Dias Belchior que, através de seu jeito simples e de seu altruísmo, conseguiu me ensinar a ser uma pessoa melhor.

AGRADECIMENTOS

A Deus, pelas conquistas conseguidas e pela minha vida.

Ao professor e amigo Arnaldo Dias Belchior, por acreditar na minha proposta de pesquisa e pela sua inestimável orientação, estando disponível e comprometido com a qualidade deste trabalho.

A Companhia de Água e Esgoto do Ceará, pelo apoio e pela ajuda na concretização deste trabalho.

Aos professores Alexandre Marcos Lins de Vasconcelos e Maria Elizabeth Sucupira Furtado pela presença na banca examinadora.

A minha mãe Glaucia Maria Barreto Nunes, pelo carinho e amor a mim dedicados e também pelos incentivos.

A minha noiva Ana Célida Veras Fernandes e a meu tio João Alberto Barreto pela dedicação e disponibilidade nas correções gramaticais deste trabalho.

Aos amigos do mestrado, pelas ajudas nos momentos mais difíceis e pelo aprendizado que me proporcionaram.

Aos demais professores do mestrado, pela constante presença e contribuições indiretas.

Ao pessoal administrativo do MIA, pela atenção e presteza sempre imediatas.

Aos especialistas em processo de desenvolvimento de software e especialistas em segurança da informação que participaram da pesquisa de campo deste trabalho.

Aos demais amigos aqui não citados, mas que certamente foram importantíssimos para a conclusão deste trabalho.

Resumo da Tese apresentada ao MIA/UNIFOR como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciência da Computação (M.Sc.)

PASS: PROCESSO DE APOIO À SEGURANÇA DE SOFTWARE

Francisco José Barreto Nunes

Setembro / 2007

Orientador: Arnaldo Dias Belchior

Programa: Ciências da Computação

A necessidade de se desenvolver produtos de software com maior qualidade promoveu a evolução da engenharia de software e exigiu que se criassem modelos e normas internacionais, voltados para a melhoria e qualidade do processo de desenvolvimento e de produtos de software. Contudo, apesar desta evolução e do investimento feito em processos de desenvolvimento ainda não há garantias de que os sistemas desenvolvidos sejam imunes a ataques ou deixem de apresentar problemas de segurança. Este trabalho propõe um conjunto de atividades de segurança que formam o Processo de Apoio à Segurança de Software (PASS) elaborado em consonância com o SSE-CMM, o OCTAVE, e com as normas ISO/IEC 27002 e ISO/IEC 15408. É apresentado o resultado de uma pesquisa de campo, realizada com especialistas nas áreas de segurança e de processo de software, sobre a importância das atividades de segurança do PASS no processo de desenvolvimento de software. Esses resultados foram avaliados através de um estudo de caso para a avaliação da aplicabilidade do PASS.

Abstract of Thesis presented to MIA/UNIFOR as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

PSSS: PROCESS TO SUPPORT SOFTWARE SECURITY

Francisco José Barreto Nunes

September / 2007

Advisor: Arnaldo Dias Belchior

Department: Computer Science

The need for developing software products with higher quality promoted the evolution of the software engineering and demanded the creation of international models and standards, related to the quality of development process and software products. However, despite this evolution and the investment made in development processes, there are still no guarantees that the developed systems are immune to attacks or do not present security problems. This work suggests a set of security activities which form the Process to Support Software Security (PSSS) in accordance with SSE-CMM, OCTAVE, and the standards ISO/IEC 27002 and ISO/IEC 15408. The results of a survey are presented, which was carried out with information security and software process specialists, on the importance of the PSSS security activities in a software development process. The results were validated by means of a case study for the appreciation of PSSS's applicability.

ÍNDICE

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

Capítulo **1**

Introdução

Este capítulo apresenta alguns aspectos de segurança de software que motivaram a realização deste trabalho.

1.1 Motivação

O investimento em desenvolvimento de software tem se tornado um fator crítico de sucesso e um risco para as organizações, que cada vez mais têm seus processos de negócios automatizados, ampliando, por conseguinte, as oportunidades de sofrerem ataques. Dessa forma, é vital para as organizações que seus produtos de software tenham qualidade e atendam às necessidades de seus clientes, assegurando a continuidade do negócio e garantindo a segurança e a integridade das informações, entre outras questões.

O Instituto Nacional de Padrões e Tecnologia do Departamento de Comércio dos Estados Unidos calculou que falhas de software representam perdas anuais para a economia norte-americana de 59,6 bilhões de dólares, incluindo o custo de ataques a códigos defeituosos (VALIM, 2004).

Segundo Davis (2004), para produzir sistemas com poucos defeitos e mais confiáveis, as práticas atuais de desenvolvimento devem mudar. Isto requer que desenvolvedores usem métodos que produzam sistemas seguros. Ou seja, sistemas seguros são aqueles que conseguem se manter funcionais mesmo sob ataque ou sob ação de algum defeito, não comprometendo a qualidade e segurança das informações. Tais métodos necessitam uma expertise em segurança por parte da organização, além da utilização de processos que produzam aplicativos seguros e com poucos defeitos. É exigido também que a organização utilize sua experiência e seus processos de segurança quando produzir, melhorar e refazer o sistema. No entanto, a definição de funcionalidades de segurança não garante que o sistema seja autenticamente seguro, pois a segurança de software é uma propriedade do sistema que emerge da totalidade de seu comportamento.

A baixa utilização de práticas para produzir produtos de software mais seguros está relacionada ao custo, às dificuldades envolvidas, ao desconhecimento e descrédito de que a segurança possa ser um diferencial na qualidade do sistema. Há ainda uma idéia errônea de

ser mais rentável e melhor aquele sistema que apresente baixo custo e que seja de desenvolvimento rápido.

Segundo McGraw (1999), há poucos recursos disponíveis para ensinar os desenvolvedores a construírem aplicativos seguros. Além disso, a pressão do mercado exige que muitos aplicativos sejam desenvolvidos de maneira muito rápida, dificultando o aprendizado sobre segurança e sobre os processos de desenvolvimento, além do envolvimento de equipes de segurança. A conseqüência dessa pressão é a ausência de qualidade do aplicativo, incluindo segurança que, às vezes, é deixada em segundo plano.

1.2 Objetivos do Trabalho

O objetivo primordial de um software seguro é a preservação da confidencialidade, integridade e disponibilidade dos ativos de informação que o sistema cria, armazena, processa, ou transmite. Além da preservação dessas propriedades, outros objetivos podem contribuir para a segurança do software (DAVIS, 2004):

- Estabelecer responsabilização dos usuários (desenvolvedores teriam maior cuidado ao conceber o software inseguro, e usuários refletiriam antes de tentar comprometer a segurança de um software caso houvesse formas de responsabilização e punição por atos indevidos e omissões).
- Implementar defesa em profundidade com múltiplas camadas de proteção.
- Garantir o não repúdio, impedindo que algum agente negue a realização de ações verdadeiramente realizadas.
- Habilitar a detecção de ataques, incluindo: permitir sua notificação, dar continuidade ao serviço, restringir o impacto, recuperar-se do dano, e prevenir ataques futuros.

Wyk (2004) formalizou algumas questões que podem contribuir para melhorar a segurança do software:

- Desenvolvedores precisam envolver a equipe de segurança o mais cedo possível no processo de desenvolvimento.
- A equipe de segurança deve auxiliar em todas as fases do processo.
- Considerar algumas táticas de instalação para que a equipe de operação possa elevar a segurança do software.
- A equipe de segurança deve aprender sobre processos e procedimentos de desenvolvimento de software da organização.

- Gaste algum tempo estudando Segurança de Software e veja quais pontos da segurança podem ser incorporados no ciclo de vida de desenvolvimento de software da organização para, apropriadamente, tratar segurança desde o início.

Uma das metas deste trabalho é auxiliar desenvolvedores de software a evitarem a abordagem de segurança "aprofundar e ajustar", em que falhas são corrigidas, quando estes aprendem ou tomam conhecimento sobre elas, atestando que a segurança não foi adequadamente considerada.

Este trabalho objetiva contribuir para a construção de software seguro e, por conseguinte, software de maior qualidade, propondo para isto o Processo de Apoio à Segurança de Software (PASS), que foi organizado a partir do SSE-CMM (2003), do OCTAVE (ALBERTS ET AL., 2001), da ISO/IEC 15408 (2005a, 2005b, 2005c), e da ISO/IEC 27002 (2005).

O SSE-CMM é um modelo que determina o processo de maturidade de uma organização que trabalha com engenharia de segurança.

O OCTAVE é um processo auto-orientado de avaliação de riscos para segurança por meio da análise dos ativos, ameaças, vulnerabilidades, e impacto organizacional.

A ISO/IEC 15408 propõe um conjunto de critérios para avaliar a segurança de produtos de tecnologia da informação, abrangendo a segurança no ambiente de desenvolvimento e a segurança da aplicação desenvolvida.

A ISO/IEC 27002 orienta para a preservação da confidencialidade, da integridade e da disponibilidade das informações, por meio da implementação de controles.

1.3 Organização do Trabalho

Este trabalho está organizado em oito capítulos, inclusive esta introdução como Capítulo 1, trazendo ainda sete apêndices, como descrito a seguir.

No Capítulo 2, **Enfoques sobre Segurança da Informação**, apresentam-se vários conceitos e enfoques sobre a segurança da informação, incluindo alguns aspectos de segurança que a própria Engenharia de Software aborda.

No Capítulo 3, **Abordagens de Segurança da Informação**, apresenta-se uma visão geral sobre as abordagens de segurança da informação utilizadas.

No Capítulo 4, **Trabalhos Relacionados a Processos Seguros**, discorre-se sobre trabalhos e artigos relacionados com o foco da proposta deste trabalho, que é promover um Processo de Apoio à Segurança de Software. Algumas características encontradas nos trabalhos serviram de complementação para as atividades e tarefas do processo proposto.

No Capítulo 5, **Importância das Atividades de Segurança no Processo de Apoio à Segurança de Software**, são analisados os resultados de uma pesquisa de campo realizada com especialistas em processo de desenvolvimento de software e especialistas em segurança da informação.

No Capítulo 6, **Processo de Apoio à Segurança de Software (PASS)**, propõe-se um processo para apoiar o desenvolvimento de software mais seguro, o qual contém um conjunto de atividades relacionadas à segurança da informação e à engenharia de segurança.

No Capítulo 7, **Aplicação do Processo de Apoio à Segurança de Software**, estão os principais resultados obtidos pela aplicação do PASS em um projeto de software real.

No Capítulo 8, **Conclusão**, encontram-se as principais conclusões deste trabalho e perspectivas futuras.

No Apêndice 1, está a tabela representando o resultado principal do mapeamento das abordagens de segurança da informação.

No Apêndice 2, está o questionário utilizado na pesquisa de campo entre os especialistas em processo de desenvolvimento de software e segurança da informação.

No Apêndice 3, é apresentado o artefato "Plano de Segurança".

No Apêndice 4, é apresentado o artefato "Relatório de Vulnerabilidade de Segurança".

No Apêndice 5, é apresentado o artefato "Relatório de Ameaças de Segurança".

No Apêndice 6, é apresentado o artefato "Relatório de Impactos de Segurança".

No Apêndice 7, é apresentado o artefato "Relatório de Exposição aos Riscos de Segurança".

Capítulo **2**

Enfoques sobre Segurança da Informação

Este capítulo apresenta vários enfoques sobre a segurança da informação.

2.1 A Segurança da Informação

Segundo Dias (2000), a segurança da informação é uma atitude, uma filosofia a ser difundida por toda a organização. Ela é muito ampla e seus objetivos envolvem a proteção de informações, variando de acordo com o tipo de ambiente computacional e com a natureza do sistema.

A segurança da informação ratifica, cada vez mais, sua participação nas empresas através da figura dos administradores de segurança, que a gerenciam, e em instituições de pesquisa e universidades através de pesquisadores, que estudam sua influência sob vários aspectos organizacionais.

A segurança da informação também busca avaliar a forma como as aplicações são criadas por causa da necessidade de garantir a confidencialidade, integridade e disponibilidade das informações processadas. Isto se dá através da verificação da inclusão de controles no processo de desenvolvimento de *software*. Controles são análises executadas com a finalidade de garantir que determinadas atividades e especificações sejam eficazmente cumpridas e que não se desviem de padrões pré-estabelecidos.

Atualmente, a busca por entender conceitos e processos inerentes à segurança da informação deve-se ao substancial crescimento de ocorrências de ataques, fraudes eletrônicas e evidências de vulnerabilidades dos sistemas de informação, suas conseqüências e a percepção de que a segurança da informação é um agente estratégico na consecução de melhoria e aumento da qualidade dos processos organizacionais. Os pilares básicos da segurança da informação incluem (DIAS, 2000):

- *Confidencialidade*: garantia de que as informações do sistema somente serão acessadas por usuários autorizados, segundo seu perfil.
- *Integridade*: garantia de que uma informação não foi alterada sem consentimento ou conhecimento de seu proprietário.
- *Disponibilidade*: capacidade de proteção da informação de tal modo que ela não seja degradada e tornada indisponível.

Além dos pilares acima citados, há outros requisitos que merecem destaque (DIAS, 2000):

- *Confiabilidade*: garantia do sistema atuar conforme o esperado.
- *Autenticação*: garantia de que a entidade é mesmo quem alega ser.
- *Não repúdio*: capacidade do sistema provar a origem e a autoria de uma ação.
- *Legalidade*: garantia de obediência à determinada legislação.
- *Privacidade*: capacidade do sistema manter incógnito um usuário.
- *Consistência*: garantia do sistema atuar de acordo com as expectativas dos usuários.
- *Auditoria*: capacidade de proteger sistemas contra erros e atos maliciosos cometidos por usuários, registrando ações executadas a fim de detectar fraudes ou atividades ilícitas.

A segurança da informação apresenta um conjunto de termos que são comuns à atividade de implantar ou avaliar segurança (ALBUQUERQUE, 2002):

- *Ataque*: problema de segurança causado por um agente a fim de obter algum ativo de valor ou causar dano àquele ativo. O retorno do ataque pode ou não ser financeiro.
- *Ativo*: algo de valor resguardado pelo sistema envolvendo, às vezes, algum objetivo de segurança. Esta informação pode ter um grande valor para determinadas entidades.
- *Vulnerabilidade*: todo ativo possui uma ou mais vulnerabilidades (pontos fracos), que podem ser exploradas gerando um ataque. Um ponto fraco pode ser um erro no código ou uma falha de especificação de segurança.
- *Ameaça*: envolve um ataque potencial e está relacionada com o conjunto de três elementos: agente (atacante), a vulnerabilidade, e o ativo de valor.

Há grande variedade de métodos de ataques via Internet e inúmeras proteções que precisam ser empregadas para prevenir ou reduzir a severidade do ataque. Segundo Lavery (2001), os ataques podem ser classificados como "passivo" e "ativo", de acordo com o objetivo do ataque. Ataques passivos não causam dano físico aos dados. Ataques ativos, por outro lado, causam danos aos dados. Salienta-se que problemas indiretos causados por um ataque passivo podem ser relevantes. Por exemplo: *Qual será o resultado de se ter uma informação confidencial lida indevidamente*?

As proteções podem ser consideradas controles e fornecem apoio para prevenir, detectar, ou se recuperar de um ataque. Lavery (2001) afirma que categorias de ataque podem ser combinadas e usadas em um único ataque.

Outros dois termos devem ser conhecidos: incidentes e impacto. Incidentes estão diretamente relacionados aos ativos, descrevendo o efeito das ameaças nos ativos. São eles que causam os impactos nos negócios.

Impacto está relacionado com a organização, sua missão e objetivos de negócio. É importante avaliar a amplitude e a gravidade do impacto. Por exemplo, o incidente da perda de um documento é pequeno, mas se o documento for secreto pode causar um grande dano ou impacto. A ISO/IEC 15408-1 (2005a) resume muito bem os conceitos de segurança e seus relacionamentos através da Figura 2.1.

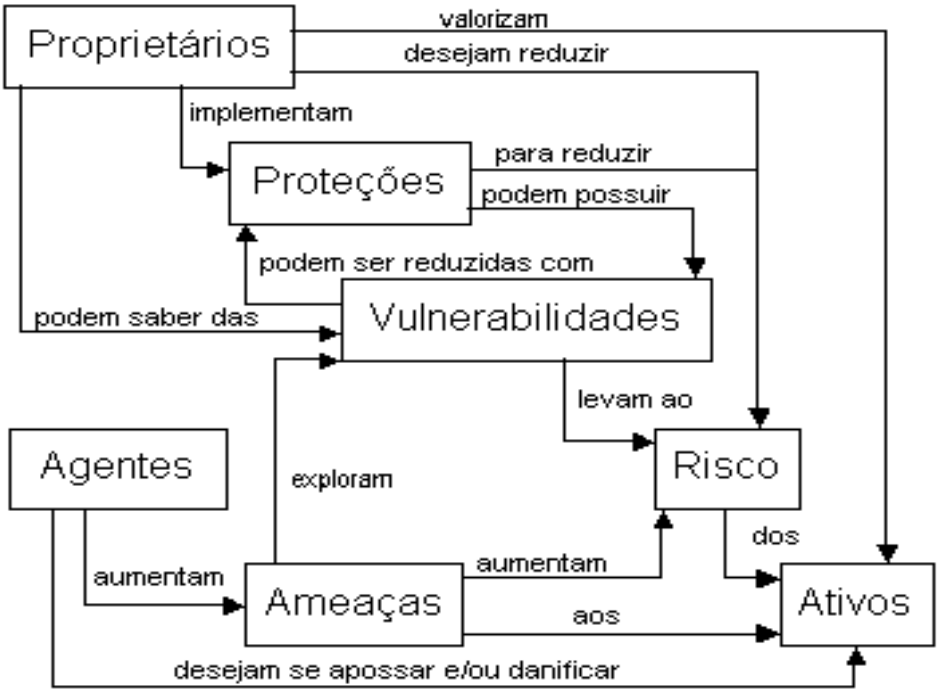

Figura 2.1: Relacionamentos e conceitos de segurança (ISO/IEC 15408-1, 2005a)

As preocupações da segurança devem incidir, principalmente, naqueles ativos cuja ratificação de ataque causaria um grande dano à organização e cuja probabilidade de ocorrência seja alta. Na Figura 2.2, percebemos que os controles de segurança devem ser primeiro aplicados nos ativos da área 1. Em seguida, implantamos controles de segurança nos ativos das áreas 2 e 3. É provável que nada precise ser feito nos ativos da área 4.

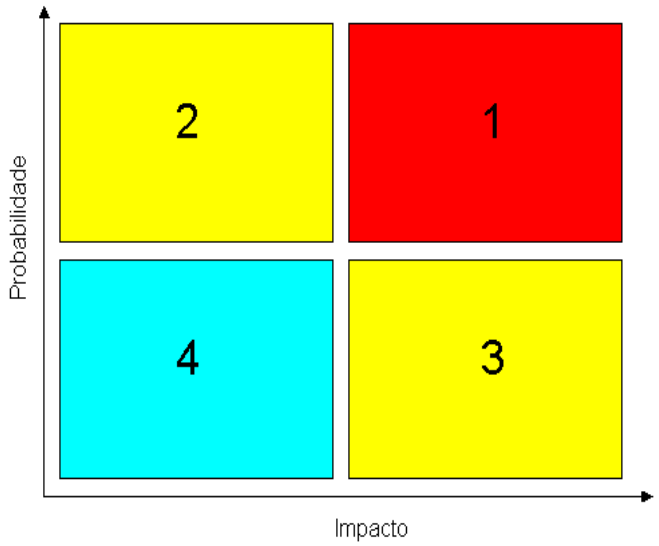

Figura 2.2: Probabilidade versus impacto (PMBOK)

2.2 A Engenharia de Segurança

A engenharia de segurança é aplicada para desenvolver, integrar, operar, gerenciar, manter, entregar e promover a evolução de sistemas, aplicações, e produtos com foco na integração de aspectos de segurança. Esses aspectos devem ser tratados na definição, gestão, e reengenharia das empresas e seus processos de negócio.

Segundo Sheard (2003), engenharia de segurança é uma prática de construir sistemas robustos em face à segurança, focando em ferramentas, tecnologias, métodos e processos requeridos para projetar, implementar, e testar sistemas, além de aumentar a segurança dos sistemas existentes para operar em ambientes altamente conectados. É um campo multidisciplinar que envolve muitas áreas de conhecimento, como segurança da informação, criptografia, engenharia de *software*, métodos organizacionais, auditoria, entre outros.

Alguns objetivos da engenharia de segurança incluem (SSE-CMM, 2003):

- Entender os riscos de segurança da empresa.
- Definir um conjunto de necessidades de segurança baseado nos riscos identificados.
- Transformar necessidades de segurança em procedimentos de segurança a serem integrados nas atividades de outras disciplinas envolvidas em um projeto e em descrições de operação ou configuração de um sistema.
- Construir confiança ou garantia na correção e eficácia dos mecanismos de segurança.
- Estabelecer que vulnerabilidades residuais de segurança de um sistema ou de sua operação geram impactos operacionais toleráveis.

- Organizar a integração das disciplinas de engenharia e demais especialidades em um entendimento comum em relação à confiança do sistema.

As atividades da engenharia de segurança são praticadas durante todas as etapas do ciclo de vida, incluindo concepção, desenvolvimento, produção, utilização, suporte, e descontinuidade (ISO/IEC 15288, 2002).

Segundo Schumacher (2001), o resultado de um sistema elaborado através da engenharia de segurança deve ser uma arquitetura que garanta aspectos de segurança tais como privacidade, confidencialidade, integridade e disponibilidade.

Contudo, freqüentes ataques e falhas de segurança demonstram que se necessita de mais subsídios para garantir a segurança em sistemas de informação. Alguns fatos atestam as deficiências em segurança nos sistemas de segurança, como, por exemplo, a infecção por programas cavalo de tróia ou por programas invasores chamados *Spyware*, que roubam ou usam o computador para invadir outras instituições.

Os fatores humanos podem comprometer a segurança das seguintes maneiras (SCHUMACHER, 2001):

- Implantar a engenharia de segurança, sendo realizada por pessoas não especializadas.
- Pessoas tentam encontrar uma solução estruturada de algum problema através de abordagens *ad hoc* e inseguras.
- Desconhecimento de que segurança é uma área complexa com múltiplas dependências.
- Desconhecimento de que segurança é altamente dependente do tempo, pois uma característica de segurança hoje pode ser insegura amanhã.

As atividades da engenharia de segurança interagem com muitas outras disciplinas, incluindo a engenharia de sistemas, engenharia de *software*, engenharia de comunicação, engenharia de hardware, engenharia de teste, administração de sistema, entre outras.

As atividades da engenharia de segurança devem ser coordenadas com muitas entidades externas, uma vez que a garantia e a aceitabilidade dos impactos operacionais residuais são estabelecidos em conjunto com o desenvolvedor, integrador, adquirente, usuário, avaliador, e demais grupos. Estas inter-relações e a interação de requisitos através de um amplo conjunto de organizações fazem da engenharia de segurança uma disciplina complexa e diferente de outras disciplinas de engenharia.

A engenharia de segurança envolve muitas disciplinas especializadas da segurança da informação, objetivando atingir resultados mais eficientes e eficazes na execução do trabalho, como exemplifica a lista abaixo (SSE-CMM, 2003):

- *Segurança de operações*: envolve a segurança do ambiente de operação e a manutenção de uma operação segura.
- *Segurança da informação*: relaciona-se com a informação e manutenção da segurança da informação durante sua manipulação e processamento.
- *Segurança de rede*: envolve a proteção do hardware, software e protocolos de rede, incluindo a informação transmitida nas redes.
- *Segurança física*: foca na proteção de prédios e locais físicos.
- *Segurança de pessoas*: relaciona-se com pessoas, sua confiança e seu entendimento sobre assuntos de segurança.
- *Segurança administrativa*: relaciona-se com aspectos administrativos de segurança e segurança em sistemas administrativos.
- *Segurança em computador*: lida com a segurança dos dispositivos de computação de todos os tipos.

2.3 Engenharia de Software Seguro

A engenharia de software é a aplicação de um enfoque sistemático, disciplinado e quantificável, para o desenvolvimento, a operação e a manutenção de software. Podem ser elencados alguns aspectos da engenharia de software, quando se busca o desenvolvimento de software seguro, como, por exemplo: a proteção, a confiabilidade, a construção de especificações formais, a verificação e validação, o desenvolvimento de sistemas críticos.

2.3.1 Proteção

Entende-se proteção de sistema como a avaliação da resistência do sistema contra ataques, acidentais ou intencionais. Há três tipos de danos causados por ataques (SOMMERVILLE, 2003):

- *Interromper serviço*: o sistema é colocado em um estado onde sua operação fica indisponível.
- *Corromper informações*: as informações e componentes do sistema são alterados inadvertidamente.
- *Revelar informações confidenciais*: informações consideradas confidenciais são expostas a pessoas não autorizadas.

Por outro lado, podem-se utilizar algumas abordagens para proteger um software:

- *Evitar vulnerabilidades*: o software é planejado objetivando a não ocorrência de vulnerabilidades.
- *Detectar e impedir ataques*: o software é planejado objetivando detectar vulnerabilidades e impedir que elas o exponham.
- *Minimizar a exposição*: o software é planejado objetivando que as conseqüências de um ataque de sucesso sejam limitadas.

Finalmente, três estratégias são importantes para aumentar a proteção do sistema: (*i*) resistência a ataques; (*ii*) reconhecimento de que está sendo atacado; e (*iii*) recuperação de danos provocados pelo ataque.

2.3.2 Confiança

A confiança em um produto de software é uma característica do software relacionada com a sua integridade, isto é, o nível de confiança dos usuários na utilização sem falha do sistema.

Para a engenharia de software, há quatro dimensões principais de confiança (SOMMERVILLE, 2003):

- *Disponibilidade*: probabilidade de que o sistema continue operando a qualquer momento.
- *Confiabilidade*: probabilidade de que o sistema funcione corretamente de acordo com as expectativas do usuário.
- *Segurança*: probabilidade de que o sistema não provoque danos aos clientes e ao ambiente.
- *Proteção*: probabilidade de que o sistema resista a tentativas de invasão deliberada ou acidental.

Aumentar a confiança de um sistema pode elevar consideravelmente o valor gasto no desenvolvimento. Além disso, aumentar o nível de confiança pode implicar reduzir o desempenho do sistema.

Sommerville (2003) afirma que a confiança é um atributo mais importante que o desempenho em alguns aspectos:

- Geralmente, os sistemas não confiáveis não apresentam segurança e, por conseguinte, não são utilizados.
- Os custos de falha de um sistema podem ser enormes.

- Um sistema que não é confiável é mais difícil de passar por melhorias, uma vez que a falta de integridade tende a estar distribuída por todo o sistema.
- Freqüentemente, é possível compensar a falta de desempenho do sistema.
- Sistemas não confiáveis podem causar a perda de informações.

De acordo com McGraw (2006), pouco se fez para classificar e categorizar problemas de segurança de software. Nesse contexto, ele propõe que a reintrodução das terminologias de defeito, de erro (*bug*), e de falha (*flaw*) possa esclarecer esse problema de categorização:

- Defeito: São vulnerabilidades geradas no projeto e na implementação que podem permanecer inertes e ter conseqüências graves.
- Erro: É um problema de implementação de software que pode existir no código e nunca ser executado.
- Falha: É um problema mais sutil instanciado no código do software ou em projeto. Cita-se como exemplo o problema de compartimentalização.

Segundo Sommerville (2003), defeitos de sistema não resultam em erros de sistema, porque o estado defeituoso pode ser transitório e corrigido antes da ocorrência de um comportamento errôneo. Pode-se afirmar também que erros em sistemas não provêm de falhas do sistema, porque o comportamento pode ser transitório, sem efeitos perceptíveis, ou o sistema pode implementar alguma rotina que descubra e recupere-o de um estado errôneo. Há três tratamentos que podem ser incluídos em um sistema para torná-lo mais confiável:

- *Evitar defeitos*: técnicas que diminuem a ocorrência de erros ou os descubram em tempo hábil antes de gerarem defeitos nos sistemas.
- *Detectar e excluir defeitos*: técnicas de verificação e validação mais eficazes no descobrimento de defeitos antes da utilização do sistema.
- *Tolerar defeitos*: técnicas que garantam que defeitos em um sistema não causem erros em sistema ou que garantam que os erros não causem falhas.

A segurança de um sistema é alcançada quando se evita que acidentes ocorram ou que as conseqüências do acidente sejam minimizadas quando da falha do sistema. Essa condição pode ser atingida de três maneiras diferentes:

- *Evitar o perigo*: o software é planejado objetivando evitar os perigos.
- *Detectar e eliminar o perigo*: o software é planejado objetivando detectar e eliminar os perigos antes de causarem acidentes.
- *Minimizar os danos*: o software é planejado para incluir requisitos de proteção objetivando limitar os danos de um acidente.

O Quadro 2.1 apresenta um vocabulário especializado para melhor compreensão dos termos envolvidos com segurança de sistemas.

Quadro 2.1: Terminologia de segurança de sistemas (SOMMERVILLE, 2003)

Termo	Definição
Acidente (desastre)	Evento não planejado que resulte em morte, ferimento, danos à propriedade ou ao ambiente.
Risco	O risco é avaliado, considerando a probabilidade do perigo, sua gravidade e a probabilidade de que ele resulte em um acidente.
Exposição	Possível perda ou dano em um sistema de computação.
Vulnerabilidade	Fraqueza em um sistema baseado em computador, que pode ser explorada para causar perdas ou danos.
Ataque	Exploração da vulnerabilidade de um sistema.
Ameaças	Circunstâncias que têm o potencial de causar perda ou dano.
Controle	Medida de proteção que reduz a vulnerabilidade do sistema.

O sistema é considerado confiável se seu funcionamento estiver conforme sua especificação. Dessa forma, a não confiabilidade do sistema será percebida caso as expectativas dos usuários não estejam sendo atendidas por causa de problemas na especificação.

2.3.3 Especificação Formal

De acordo com Sommerville (2003), a especificação formal é uma especificação expressa em uma linguagem cujo vocabulário, sintaxe e semântica são formalmente definidas. Essa necessidade de uma definição formal significa que as linguagens de especificação precisam ter como base conceitos matemáticos, cujas propriedades tenham sido investigadas e sejam bem compreendidas.

A argumentação para a utilização de uma especificação formal e de métodos formais é que sua análise detalhada e rigorosa promova o desenvolvimento de sistemas mais seguros. Contudo, esta argumentação não se comprova devido a um conjunto de razões, entre elas (SOMMERVILLE, 2003):

- *Evolução da engenharia de software*: o aperfeiçoamento dos métodos e processos de engenharia de software resulta na melhoria da qualidade do software.
- *Prazo de entrega*: os prazos cada vez menores para entrega de produtos de software exigidos pelo mercado em detrimento até da qualidade do produto.
- *Limitações da especificação formal*: a especificação formal não é adequada para qualquer tipo de software e em toda a sua extensão.

Pressupõe-se dessas razões que o risco de se utilizar a especificação formal na maioria dos projetos de software é maior que o benefício trazido. No entanto, uma grande vantagem da especificação formal é encontrar inconsistências e erros na especificação de requisitos, pois

podem ser tratados em fase inicial do processo de desenvolvimento reduzindo custos com retrabalho. Informações adicionais acerca da especificação formal podem ser encontradas em Berard (2001) e em Monin (2003).

O ato de especificar formalmente é um processo dispendioso. Contudo, o mesmo nível de segurança ou de proteção poderia ser alcançado, a custos menores, pelo uso de outras técnicas de validação, como as inspeções e os testes de sistema (SOMMERVILLE, 2003).

2.3.4 Verificação e Validação

Em geral, o nível de segurança de um sistema depende de sua importância para a organização, a maior exigência dos usuários, e o ambiente dinâmico do mercado, daí a importância das atividades de verificação e validação.

Verificação e validação são similares, mas tratam de assuntos diferentes. A verificação confirma que os produtos de trabalho refletem as exigências especificadas para eles. A validação demonstra que o produto cumpriu ou cumprirá seu uso planejado.

Na verificação, há a confirmação de que os produtos de trabalho refletem apropriadamente os requisitos especificados para eles. Assegura que "foi construído certo o produto". Na validação, há a confirmação de que o produto fornecido irá atender ao seu uso pretendido. Assegura que "foi construído o produto certo". As técnicas de verificação e validação envolvem: inspeção, análise estática, e testes de software.

A verificação é um processo incremental porque acontece ao longo do desenvolvimento do produto e dos produtos de trabalho, começando com a verificação dos requisitos, progredindo pela verificação dos produtos de trabalho envolvidos, e culminando na verificação do produto completo.

2.3.5 Desenvolvimento de Sistemas Críticos

Para se desenvolver um software confiável, pode-se partir de dois princípios SOMMERVILLE, 2003):

- *Evitar defeitos*: as fases de projeto e de implementação do processo de desenvolvimento de software implementam ações que reduzem os erros e aumentam a descoberta de defeitos antes de o sistema estar em uso.
- *Tolerância a defeitos*: o projeto do sistema deve abordar aspectos que permitam ao sistema detectar defeitos ou comportamentos não esperados e gerenciá-los antes do defeito se concretizar. Por exemplo, na ocorrência de um defeito, o sistema isola a porção defeituosa, para que não haja comprometimento total do sistema.

Os custos para descobrir e eliminar defeitos crescem de maneira exponencial, pois ao passo que o software se torna mais confiável é preciso testes adicionais para encontrar menos defeitos.

Para Sommerville (2003), existem quatro aspectos de tolerância a defeitos:

- *Detecção de defeitos*: o sistema deve detectar que uma determinada combinação de estados ocorreu, o que pode levar a uma falha do sistema.
- *Avaliação de danos*: as partes do sistema afetadas pelo defeito devem ser detectadas.
- *Recuperação de defeitos*: o sistema precisa ser restaurado a um estado 'seguro' conhecido. Isso pode ser alcançado corrigindo-se o estado afetado (recuperação de erro para frente) ou restaurando-se o sistema a um estado 'seguro' conhecido (recuperação de erro para trás).
- *Reparo de defeitos*: envolve modificar o sistema, de modo que o defeito não se repita. Em muitos casos, os defeitos de software se manifestam como estados transientes. Eles se devem a uma peculiar combinação de entradas do sistema. Nenhum reparo é necessário, uma vez que o processamento normal pode reassumir imediatamente depois da recuperação do defeito.

Existe uma abordagem complementar que pode ser utilizada para implementar tolerância a defeitos em software chamada de Programação Defensiva (SOMMERVILLE, 2003). Nessa abordagem, os programadores supõem que pode haver defeitos não detectados ou inconsistências em seus programas. Um código redundante é incorporado, a fim de verificar o estado do sistema após modificações, e de assegurar que a mudança de estado seja consistente. Se inconsistências forem detectadas, a mudança de estado é redefinida ou o estado é restaurado para um estado correto conhecido.

A seguir será abordada a segurança de software, no contexto do processo de desenvolvimento de software seguro.

2.4 Segurança de Software

A segurança de software envolve o desenvolvimento de software seguro: projeto de software para ser seguro, garantia de software seguro, e educação de desenvolvedores, arquitetos, e usuários sobre como construir artefatos de software seguro (McGRAW, 2004).

Segurança de software é um assunto muito amplo que leva em consideração mecanismos de segurança (como controle de acesso) e projeto para segurança (como projeto robusto que dificulte ataques).

Os engenheiros de software, geralmente, não possuem treinamento em segurança de software. Além disso, poucos são os engenheiros que vivenciam situações relacionadas com segurança. Por conseguinte, requisitos relacionados com segurança são inadequadamente especificados ou até não compreendidos resultando em sistemas vulneráveis e com falhas (SHEARD, 2003).

Sheard (2003) afirma que corrigir sistemas com atualizações (*patches*) ou quaisquer alternativas de correção posteriores à entrega do produto são muito custosas, tanto financeiramente quanto para a imagem da instituição desenvolvedora. Logo, engenheiros de software necessitam conhecer acerca da segurança de software. Pode-se afirmar que as estratégias de segurança de software, em geral, envolvem entender o valor do sistema para o negócio, analisar ameaças, identificar e reduzir riscos.

Quando se fala em segurança de software, os desenvolvedores devem ter consciência de que todas as entidades externas (sistemas, usuários, etc.), que interagem com o sistema, são consideradas não confiáveis obrigando sempre que sejam realizadas autenticações e validações antes de executar operações com dados de entrada (SHEARD, 2003).

Por conseguinte, o recomendável é que todas as rotinas e funções devam ser executadas sob um perfil de acesso com um mínimo de direito para limitar danos potenciais, caso a rotina ou função esteja comprometida. Isto é chamado de menor privilégio (*least privilege*).

Deve-se considerar fortemente que todo sistema é passível de falhas. Contudo, devemos garantir que, em caso de falha, informações consideradas críticas não fiquem desprotegidas. Portanto, não devemos mostrar nas mensagens de erro informações que possam comprometer a segurança do sistema. Por exemplo: indicar que a senha está errada em vez de omitir o que está errado, senha ou usuário.

Alguns aspectos devem ser levantados quando se fala em desenvolver software seguro:

- Escrever código inseguro simplesmente pela não realização de ações, como a ausência de textos explicativos no código, funções inseguras, e uso de bibliotecas não padronizadas.
- Não há como garantir software 100% seguro e confiável. Contudo, pode-se reduzir o risco a níveis aceitáveis.
- A complexidade e interconexão dos sistemas são crescentes. Isso pode resultar em sistemas mais inseguros.

- O gerenciamento de riscos do software norteia onde os recursos devem ser adequadamente aplicados, uma vez que mostra os pontos de maior impacto negativo.
- Realizar auditorias.
- Ter um projeto voltado para segurança.
- Realizar testes de segurança.

A segurança deve ser entendida como investimento em vez de um custo uma vez que pode agregar valor. Se as necessidades de segurança não estiverem bem definidas e alinhadas com os objetivos definidos para um sistema, o investimento pode virar custo, que pode representar muito nos recursos de uma organização. Como resultado, é melhor avaliar o valor das medidas de segurança e compará-las com uma avaliação das conseqüências financeiras potenciais de um ataque. Expectativas de usuários também contribuem para com as medidas de segurança empregadas no sistema (LAVERY, 2001).

2.4.1 Conceitos e Princípios da Segurança de Sistema

Alguns conceitos gerais sobre segurança de sistema incluem: (SHEARD, 2003)

- *Autenticação*: identificação positiva de entidades que acessam o sistema. Requer uma função que troque informações de identificação entre as entidades comunicantes. Não se pode confiar apenas na autenticação como proteção para garantir a segurança de um sistema contra acessos não autorizados. Por isso, deve-se combinar a autenticação com requisitos de identificação e autorização, para aumentar a segurança no acesso de um sistema. Ou seja, primeiramente, um usuário se identifica em um sistema, depois o sistema autentica ou não o usuário baseado na identidade. Finalmente, estando o usuário autenticado, o sistema autoriza que o usuário realize determinadas ações no sistema baseado em suas características.
- *Autorização*: define aquilo que as entidades autenticadas têm direito a acessar e fazer no sistema. Seu objetivo é prevenir que entidades não autorizadas tenham acesso a informações confidenciais ou executem serviços restritos.
- *Confidencialidade*: garante que uma informação sensível não seja compreendida por entidades desautorizadas. Para garantir confidencialidade, técnicas criptográficas são empregadas para proteger informações sensíveis dispostas em meios não protegidos.

- *Integridade*: proteção dos dados transmitidos entre partes, garantindo que esses dados permaneçam intactos sem sofrer alteração.
- *Não repúdio*: mecanismo que relaciona uma entidade a uma ação ou operação. Fornece uma evidência inegável de que uma determinada ação ocorreu, precavendo-se contra possível negação da execução dessa ação.
- *Auditoria*: "exame independente de um trabalho, produto, ou conjunto de produtos de trabalho para avaliar a aderência a especificações, padrões, contratos ou demais critérios" (IEEE, 1990). Envolve também o registro de atividades que ocorrem no sistema (produto) ou entre sistemas através de uma trilha de auditoria, gerando o chamado *log* de auditoria.
- *Política*: conjunto de regras ou restrições que obrigam, proíbem, ou limitam o escopo das ações realizadas por entidades ou limitam o acesso a certos recursos ou serviços do sistema.
- *Privacidade*: significa que uma entidade controla o uso da informação, protegendo-a contra outros acessos.
- *Confiança*: refere-se ao total de mecanismos de segurança implementados em um sistema.

Apresentam-se a seguir os dez princípios para desenvolver software seguro, identificados por McGraw (2000b) que objetivam isolar os pontos mais importantes, quando se constrói um sistema seguro:

Princípio 1: *Proteger o elo mais frágil*.

As partes mais frágeis ou desprotegidas de um sistema são as mais suscetíveis a um ataque. Mesmo que se aplique um esforço semelhante em todas as partes do sistema, é provável que se encontrem problemas nas partes que mais precisam de melhorias.

Identificar os componentes mais frágeis torna-se mais simples, se for realizada uma análise de riscos. Deve-se tratar primeiro tudo o que for considerado de maior risco. Esta estratégia deve ser realizada repetidamente, pois a segurança nunca é uma garantia.

Princípio 2: *Defesa em profundidade*.

A idéia é gerenciar o risco com diversas estratégias de defesa de tal forma que, se uma camada de defesa sucumbir, haverá outra camada para prevenir ou barrar o ataque (McGRAW, 2000c).

Princípio 3: *Falha segura*.

Problemas de segurança relacionados a falhas devem ser evitados. O problema é quando muitos sistemas falham de alguma forma, eles têm um comportamento inseguro. Em tais sistemas, atacantes apenas precisam causar o tipo certo de falha ou esperar para que o tipo certo de falha ocorra (McGRAW, 2000c). Logo, em uma situação de falha, o sistema deve manter-se seguro de forma a não permitir a exploração dessa falha.

Princípio 4: *Menor privilégio*.

Este princípio declara que apenas o menor acesso necessário para realizar uma operação deveria ser atribuído, e o acesso deveria ser atribuído apenas no menor período de tempo necessário (McGRAW, 2000d).

Princípio 5: "*Compartimentalização*".

A idéia básica é que se pode examinar o dano total causado a um sistema, caso seja possível separá-lo em diversas unidades isoladas (McGRAW, 2000d).

Princípio 6: *Simplicidade*.

A complexidade aumenta o risco de problemas. Projetos complexos são difíceis de compreender, e são mais propensos a terem problemas imperceptíveis. Um código complexo é mais difícil de analisar e manter, além da possibilidade de possuir mais defeitos (McGRAW, 2000e).

Princípio 7: *Promover privacidade*.

Usuários geralmente consideram a privacidade como uma preocupação de segurança. Não se deve fazer algo que comprometa a privacidade do usuário. Deve-se ser o mais diligente quanto possível na proteção de qualquer informação pessoal que o usuário forneça (McGRAW, 2000e).

Princípio 8: *É difícil manter segredos.*

Usuários não desejam que seus dados pessoais sejam divulgados indevidamente. Logo, chaves criptográficas devem ser mantidas em segredo com o fito de evitar falsificação ou escuta ilegal (McGRAW, 2000f).

Princípio 9: *Não estenda facilmente sua confiança*.

Não se pode confiar nos usuários e nem nos servidores (McGRAW, 2000f).

Princípio 10: *Confie na comunidade*.

O uso contínuo sem falha favorece a confiança (McGRAW, 2000f). Isto é, a falta de comunicados sobre falhas de segurança de determinados produtos pode indicar que os produtos são seguros.

2.4.2 Segurança no Desenvolvimento de Software

As preocupações básicas quando se fala de segurança em desenvolvimento de software são: (*i*) Segurança do sistema a ser desenvolvido; e (*ii*) Assegurar que a segurança da aplicação foi conseguida após seu desenvolvimento (ALBUQUERQUE, 2002).

2.4.2.1 Segurança do Sistema a Ser Desenvolvido

Muitas questões ainda estão sem respostas, quando pensamos em uma forma de produzir eficazmente um sistema mais seguro (ALBUQUERQUE, 2002):

- *Como descobrir os requisitos de segurança da aplicação e especificar as necessidades de segurança do cliente*?
- *Como mapear os requisitos de segurança em especificações do sistema*?
- *Como projetar e implementar essas especificações*? e
- *Como testar validando a segurança do sistema*?

Há muitas recomendações sobre práticas para melhorar segurança e sobre programação segura. Provavelmente, uma vez que se pratiquem tais recomendações, muitas falhas de segurança nos sistemas podem ser eliminadas (Capítulo 4).

2.4.2.2 Assegurar a Segurança da Aplicação

Assegurar a segurança da aplicação não envolve apenas elaborar um sistema mais seguro, mas também permitir que os clientes e envolvidos se convençam disto. Portanto, é importante a participação do cliente e do usuário na definição de cada atributo de segurança.

Uma maneira de demonstrar a segurança do sistema para o cliente é através de testes. Assim, o cliente e demais envolvidos podem acompanhar os testes realizados pela equipe interna ou equipe independente, bem como verificar seus resultados.

A segurança nas aplicações é necessária por duas razões:

- *Aspectos legais*: normas internas e políticas de segurança que devem ser obedecidas; e
- *Ameaças aos sistemas e aos ativos de informação*: as ameaças que têm efeito direto sobre o negócio devem ser eliminadas ou reduzidas.

2.4.3 O Comportamento Inesperado

O comportamento inesperado de defeito de segurança é um estado inapropriado do programa ocasionado por uma vulnerabilidade do programa. Segundo Pfleeger (2002), uma vulnerabilidade descreve uma classe de falhas, como uma sobrecarga de memória (*buffer overflow*),

que ocorre quando muitos dados são escritos em um espaço de armazenamento de tamanho fixo, tentando sobrecarregar o espaço de armazenamento.

"*Bug*" em software é um termo que pode significar diferentes coisas. Dependendo do contexto, um "*bug*" pode ser um erro na interpretação de um requisito, um erro de sintaxe em um pedaço de código malicioso, ou a causa de uma parada no sistema.

Ainda não há técnicas para eliminar ou tratar todas as falhas de segurança do programa. Há duas razões para isto:

- Controles de programa se aplicam em nível do programa individual e programador. Isto é, controles só são especificados em relação à funcionalidade esperada do programa e entendida pelo programador, sem contextualizar as vulnerabilidades, ameaças e riscos de segurança.
- Técnicas de programação e de engenharia de software mudam e evoluem mais rapidamente do que as técnicas de segurança de computador.

O comportamento inesperado pode ser causado também por dois tipos de códigos maliciosos bem conhecidos: Armadilha (*Trapdoor*) e Canal oculto (*Covert channel*). Uma *armadilha* é um ponto não documentado de entrada no sistema. É inserido durante a implementação, talvez para testar alguma parte do sistema, ou para facilitar futuras modificações, ou para permitir acesso caso um módulo do sistema falhe, entre outras coisas.

É importante lembrar que uma falha não está diretamente ligada à armadilha, que pode ser uma técnica muito útil para teste, correção e manutenção de programa. No entanto, a falha está no processo de desenvolvimento, que não garante que uma armadilha esteja fechada, quando não mais for necessária. Isto é, a armadilha torna-se uma vulnerabilidade se ninguém a perceber ou agir para prevenir e controlar seu uso em determinadas situações.

O *canal oculto* refere-se aos programas que comunicam informação a pessoas que não deveriam recebê-la. A comunicação "viaja" despercebida, acompanhando outra comunicação devida. Isto é um meio de extrair dados clandestinamente.

É largamente conhecido que é melhor focar em prevenção do que em remediação. Isto é conseguido com a implantação de controles durante o desenvolvimento de software, para encontrar e eliminar quaisquer tipos de falhas (PFLEEGER, 2002).

2.4.4 Segurança do Produto e do Processo

Pfleeger (2002) afirma que tanto o produto quanto o processo podem contribuir para a qualidade e, em particular, para a segurança como um aspecto da qualidade. Para a segurança do produto, podemos elencar entre outras boas práticas: a modularidade, e o encapsulamento.

Do ponto de vista da segurança, a *modularidade* deveria melhorar a probabilidade de uma implementação ser correta. Pequenas porções, componentes ou módulos de um sistema, é um fator importante que ajuda os analistas de segurança a perceberem o que cada porção ou componente do sistema faz. Os analistas de segurança devem entender cada componente como uma unidade independente e ter certeza do seu efeito em outros componentes.

O *encapsulamento* esconde detalhes da implementação de um componente. Mesmo assim, os componentes devem compartilhar informação com outros. Contudo, aconselha-se que este compartilhamento seja documentado de tal forma que se conheçam os meios que um componente afeta outros componentes do sistema.

Para a segurança do processo, podemos elencar entre outras boas práticas de segurança: a revisão por pares (*peer review*), e a análise de perigos. A *revisão por pares* pode ser classificada em três tipos (PFLEEGER, 2002):

- *Revisão*: o artefato é apresentado informalmente a uma equipe de revisores; o objetivo é ter o consenso antes que o desenvolvimento prossiga.
- *Walkthrough*: o artefato é apresentado à equipe pelo seu desenvolvedor que lidera e controla a discussão. Educação é o objetivo, e o foco é no aprendizado de um único documento.
- *Inspeção*: é um processo mais formal com uma análise detalhada, em que o artefato é verificado em relação a uma lista de itens. O desenvolvedor não lidera a discussão e a identificação e correções de falhas são controladas por medidas estatísticas.

Durante o processo de revisão, é importante manter a rastreabilidade do que cada revisor descobre e quão rápido ele descobre. Este registro sugere se revisores precisam de treinamento e se certos tipos de falhas são mais difíceis de encontrar. Adicionalmente, uma análise da origem da causa para cada falha encontrada pode revelar que a falha poderia ter sido descoberta antes no processo. O registro de falhas pode também ser usado para construir um *checklist* de itens a serem verificados em revisões futuras.

A *análise de perigo* é um conjunto de técnicas voltadas para expor estados de perigo de um sistema. Em particular, pode ajudar a expor problemas de segurança e a identificar estratégias de prevenção e redução para tratar desses problemas. Isto é, a análise de perigo encontra causas prováveis de problemas de tal forma que se possa aplicar uma técnica apropriada, para prevenir o problema ou amenizar suas conseqüências. A análise de perigo envolve o desenvolvimento de listas de perigo, bem como procedimentos para explorar cenários hipotéticos para considerar perigos não óbvios.

2.4.5 A Construção de um Bom Projeto de Software

Segundo Pfleeger (2002), algumas atividades são úteis na construção de software seguro:

- realizar prognóstico e gerência de risco;
- usar uma filosofia de tolerância a falhas;
- ter uma política consistente de manipulação de erros;
- capturar a descrição e histórico do projeto; e
- usar padrões de projeto.

Durante o prognóstico, tenta-se prever os riscos envolvidos na construção e utilização do sistema. Postulam-se quais eventos indesejáveis podem ocorrer e elaboram-se planos para evitá-los ou, no mínimo, mitigar seus efeitos. Prognóstico e gerência de risco são importantes para segurança onde sempre se lida com eventos prejudiciais de conseqüências negativas. Os prognósticos ajudam a decidir quantos e quais controles utilizar.

Projetos deveriam antecipar falhas e manipulá-las de forma a minimizar sinistro e maximizar segurança. Deseja-se que o sistema seja livre de falhas e que reaja de forma aceitável a uma falha. Isto é chamado de detecção ativa de falha. Na verdade, deve-se assumir que o sistema falhará, por isso, deve-se ter certeza de que a falha inesperada não cause a parada do sistema, destrua dados, ou acabe com vidas.

Se corrigir uma falha for arriscado, inconveniente, ou caro, pode-se escolher, em vez disso, praticar a tolerância a falhas: isolar o dano causado pela falha e minimizar sinistro aos usuários.

Pode-se projetar ou codificar de modo defensivo pela construção de uma política consistente para manipular as falhas. Tipicamente, as falhas incluem:

- falha ao prover um serviço;
- prover o serviço ou dado errado; e
- corromper os dados.

Pode-se inserir no projeto uma maneira particular de manipular cada problema a partir da seleção de um dos três meios abaixo:

- *Restauração*: restaurar o sistema a um estado anterior e realizar o serviço novamente, usando outra estratégia.
- *Correção*: restaurar o sistema a um estado anterior, corrigindo alguma característica do sistema, e realizar o serviço novamente, usando a mesma estratégia.

- *Reporte*: restaurar o sistema a um estado anterior, reportando o problema a um componente de manipulação de erro, e não fornecer o serviço novamente.

A descrição e o histórico do projeto explicam os motivos pelo qual o sistema foi construído de uma maneira em vez de outra. Adicionalmente, o histórico do projeto permite verificar os padrões e saber quais entre eles são melhores em determinadas situações.

2.4.6 Aprender com os Erros

À medida que se projetam e se constroem sistemas, aconselha-se documentar as decisões. Não apenas aquilo que se decidiu fazer e o porquê, mas também aquilo que se decidiu não fazer e o porquê. Assim, após a finalização do sistema, pode-se usar a informação a cerca das falhas, e como foram encontradas e corrigidas, para dar melhor entendimento sobre o que leva às vulnerabilidades e à sua exploração.

Com esta informação, podem-se construir guias de codificação e *checklist* para ajudar os engenheiros de software. Portanto, não é necessário se repetir os mesmos erros, podendo estes ser evitados. *Checklist* e guias podem ser valiosos, especialmente durante revisões e inspeções, pois auxiliam a detectar erros típicos que causam falhas de segurança.

2.4.7 Provas de Corretude de Programa

Um especialista em segurança deseja ter certeza que um programa compute um resultado em particular, compute corretamente, e faça apenas aquilo que se supõe fazer.

A técnica Verificação de Programa, por exemplo, pode demonstrar a corretude de programas específicos. Verificação de Programa envolve fazer afirmações iniciais sobre as entradas e, depois, verificar se a saída desejada foi gerada.

Finalmente, a instrução terminal do programa é associada com a saída desejada. Aplicando-se um analisador lógico, pode-se provar que as instruções iniciais, através das implicações das instruções do programa, produzem a condição terminal. Desta forma, mostra-se que um programa em particular atinge seu objetivo.

2.4.8 Programa Confiável

Um programa para ser considerado confiável deve conduzir os testes e a análise com rigor, procurando também por certas características chaves: (PFLEEGER, 2002):

- *Corretude funcional*: o programa faz aquilo que se supõe e trabalha corretamente.
- *Aplicação de integridade*: mesmo que se apresentem comandos errados ou comandos de usuários não autorizados, o programa mantém a corretude dos dados com qual mantém contato.

- *Privilégio limitado*: ao programa é permitido acessar dados seguros, mas o acesso é minimizado e nem os direitos de acesso, nem os dados são passados a outros programas não confiáveis ou retornam a um usuário não confiável.
- *Nível apropriado de confiança*: o programa foi examinado e categorizado em um nível de confiança apropriado para o tipo de dado e ambiente no qual ele será usado.

2.4.9 Padrões de Desenvolvimento de Programas

Pode-se exercitar um nível adicional de controle sobre o desenvolvimento do software considerando os padrões de software.

Empresas preocupadas com segurança e comprometidas a seguir os padrões de desenvolvimento de software realizam auditorias de segurança. Em uma auditoria, uma equipe independente de avaliação de segurança avalia a aderência do desenvolvimento a padrões e guias. A equipe revisa os requisitos, projeto, documentação, planos e dados de teste, e código. Sabendo que há auditoria em todas as fases e artefatos do processo de desenvolvimento, um desenvolvedor não corre o risco de inserir código suspeito em um componente.

2.5 A Segurança Física

Apesar de o enfoque ser em desenvolvimento seguro de sistemas através de atividades de segurança acompanhando todo o ciclo de vida do software, é essencial esclarecer a importância da segurança física nesse contexto.

A segurança física trata das ameaças contra recursos (sistemas, mídias, códigos fonte, entre outros) e das medidas de segurança a serem executadas para proteger fisicamente estes recursos, bem como informações sensíveis.

Assim, de nada adianta implementar as atividades mais eficazes de segurança de software e treinar analistas e programadores em técnicas de desenvolvimento seguro se inexiste, por exemplo, a preocupação de se manter um controle de acesso físico à área de desenvolvimento ou se os códigos fonte de funcionalidades estratégicas de um sistema são deixados expostos sem controle de acesso lógico.

Atentar para segurança física é importante e deve ser considerado para se atingir o sucesso do desenvolvimento seguro de software.

2.6 Conclusão

O capítulo apresentou vários enfoques sobre a segurança onde se identifica alguma correlação entre os enfoques da segurança da informação e da engenharia de software.

Salienta-se que o foco dado pela segurança da informação é na garantia da integridade, da confidencialidade e da disponibilidade dos ativos de informação. Isto é, a segurança da informação orienta-se pelo contexto semântico do termo inglês "*security*".

A engenharia de software aborda a segurança por meio de um contexto diferenciado com foco na proteção. Isto é, percebe-se que a engenharia de software orienta-se pelo contexto semântico do termo inglês "*safety*".

Destaca-se que o presente trabalho relaciona-se com os aspectos de segurança da informação, considerando a integridade, a confidencialidade e a disponibilidade dos ativos de informação.

O capítulo seguinte apresentará algumas abordagens que tratam da segurança da informação e que nortearam a pesquisa.

Capítulo **3**

Abordagens de Segurança da Informação

Este capítulo apresenta uma visão geral sobre abordagens de segurança da informação.

Algumas abordagens sobre segurança da informação têm sido bastante referenciadas e são consideradas relevantes neste contexto, como: o SSE-CMM (2003), a ISO/IEC 27002 (2005), a ISO/IEC 15408 (2005a, 2005b, 2005c), e o OCTAVE (ALBERTS, 2001) de análise de riscos.

O SSE-CMM (2003) (*Systems Security Engineering – Capability Maturity Model*) é um modelo para melhorar e avaliar a capacidade da engenharia de segurança, com seus processos baseados em várias disciplinas da segurança de informação.

A ISO/IEC 27002 (2005) objetiva preservar a confidencialidade, integridade e disponibilidade das informações, através da implementação de controles, políticas, práticas ou processos. Esses controles garantem que os objetivos estabelecidos para a segurança serão atendidos satisfatoriamente. O essencial para uma organização é identificar os requisitos de segurança.

Para a ISO/IEC 15408 (2005a, 2005b, 2005c), o desenvolvimento seguro de software envolve segurança tanto do ambiente de desenvolvimento quanto da aplicação desenvolvida. As necessidades de segurança devem ser tratadas em todo o ciclo de vida, passando pela gerência de requisitos de segurança, especificação funcional, projeto de alto nível, projeto de baixo nível, até a implementação final do sistema em seu ambiente de produção.

O OCTAVE é um processo estratégico de planejamento e avaliação baseado em riscos para segurança. Os riscos dos ativos mais críticos são usados para priorizar áreas que necessitam de melhoria e para definir a estratégia da organização (ALBERTS, 2001).

3.1 O SSE-CMM

The International Systems Security Engineering Association estendeu o CMM (*Capability Maturity Model*) para o SSE-CMM (Engenharia de Segurança de Sistemas – Modelo de Maturidade e Capacidade), que veio a tornar-se a norma ISO/IEC 21827 (2002). Essa iniciativa foi uma forma de acreditar internacionalmente a eficácia da aplicação do modelo.

A capacidade do processo refere-se ao potencial de uma organização em executar uma determinada ação ou processo, sendo um intervalo dentro do qual a organização supõe operar. Maturidade do processo implica crescimento potencial na capacidade e indica tanto a eficácia na realização de uma ação ou de um processo organizacional quanto a consistência com o qual é aplicado em todas as partes da organização (HUMPHREY, 1998).

Nesse contexto, apreende que o SSE-CMM pode ser utilizado como uma proposta de melhoria contínua (por área de processo) ou uma proposta de melhoria em estágio (conjunto de áreas de processo em relação a um nível de capacidade).

O objetivo do SSE-CMM é desenvolver a engenharia de segurança como uma disciplina definida, madura e mensurável. Seu modelo e seus métodos de avaliação almejam permitir:

- Investimentos focados em ferramentas da engenharia de segurança, treinamento, definição de processos, práticas de gestão, e melhorias.
- Garantia baseada na capacidade, isto é, confiança baseada no nível de maturidade das práticas e dos processos de segurança.

O escopo e a aplicação do SSE-CMM envolvem:

- As atividades de engenharia de segurança de sistema para um produto seguro ou um sistema confiável envolvendo todo o ciclo de vida.
- Os requisitos para desenvolvedores de produto e de sistemas seguros, integradores, e organizações que fornecem serviços de computação segura e de engenharia de computação segura.
- Todos os tipos e tamanhos de organizações de engenharia de segurança da área comercial a governo, passando por centros de pesquisa.

O SSE-CMM aplica-se a uma grande variedade de organizações, que praticam a engenharia de segurança no desenvolvimento de programas de computador. O modelo contém práticas para determinar e analisar vulnerabilidade de segurança, avaliando impactos operacionais, e fornecendo informação e guias para os grupos envolvidos. Essas práticas focam o maior entendimento das necessidades de segurança por parte dos clientes através de maior interação com eles.

O SSE-CMM pode ser utilizado como:

- Uma ferramenta para avaliar as práticas da engenharia da segurança e definir melhorias.

- Um método pelo qual organizações possam ser avaliadas, estabelecendo a confiança na capacidade da organização, garantindo um sistema seguro.
- Um mecanismo padrão para clientes avaliarem a capacidade da engenharia de segurança de um fornecedor.

O SSE-CMM foi desenvolvido para evoluir a prática da engenharia de segurança com o objetivo de melhorar a qualidade, disponibilidade, e reduzindo o valor gasto na produção de sistemas seguros, e de produtos confiáveis.

O SSE-CMM afirma que a segurança é parte integrante dos esforços de engenharia que tratam dos problemas de segurança do *hardware*, *software*, sistemas ou empresa. Este modelo define características de um processo de engenharia de segurança o qual é explicitamente definido, gerenciado, medido, controlado, e eficaz. O objetivo é entrar em um contínuo ciclo de avaliação do status atual, fazendo melhorias e repetindo o ciclo. As etapas do SSE-CMM para melhorar processo seguem o modelo IDEAL (GREMBA, 1997) do SEI:

- *Iniciar*: preparar para um esforço de melhoria de sucesso.
- *Diagnosticar*: determinar onde você está em relação aonde você quer chegar.
- *Estabelecer*: planejar os detalhes de como você atingirá seu destino.
- *Agir*: fazer o trabalho de acordo com o plano.
- *Aprender*: aprender com a experiência e melhorar sua habilidade.

3.1.1 Visão Geral do Processo de Engenharia de Segurança

O SSE-CMM considera que a engenharia de segurança envolve três pilares: Risco, Engenharia, e Garantia. No nível mais simples, o processo de risco identifica e prioriza perigos inerentes ao sistema ou produto desenvolvido. O processo de engenharia de segurança trabalha com outras disciplinas de engenharia, para determinar e implementar soluções aos problemas apresentados. O processo de garantia estabelece confiança nas soluções de segurança e transmite essa confiança aos clientes. Esses três fatores trabalham juntos para assegurar que os resultados do processo da engenharia de segurança atinjam seus objetivos.

3.1.1.1 Risco

O maior objetivo da engenharia de segurança é a redução de riscos. Para se atingir um limite aceitável de redução, realiza-se uma avaliação de risco, que é um processo para identificar problemas que ainda não ocorreram.

Os riscos são avaliados examinando-se probabilidades de ameaças e vulnerabilidades além de considerar o impacto de um incidente indesejado. A probabilidade pode ser apenas prevista dentro de

certos limites. Além disso, impactos analisados para um risco em particular têm uma incerteza associada de que o incidente indesejado possa não ocorrer. Devido a este nível de incerteza, o planejamento e a justificativa de segurança podem ser dificultados. Uma maneira de lidar com esse problema é implementar técnicas para detectar a ocorrência de incidentes.

O risco de incidentes é relacionado com três componentes: ameaça, vulnerabilidade e impacto. Vulnerabilidades são propriedades de algum objeto, que podem ser exploradas por uma ameaça e incluem fragilidades. Caso a ameaça ou a vulnerabilidade não esteja presente poderá não haver incidente nem risco. A gerência de risco é o processo de avaliar e quantificar risco, e estabelecer um nível aceitável de risco para a organização. Gerenciar risco é uma parte importante da gestão de segurança.

Riscos são reduzidos através da implantação de proteções, que podem lidar com a ameaça, a vulnerabilidade, o impacto, ou o risco. Contudo, não é possível reduzir todos os riscos ou mitigar completamente qualquer risco em particular. Isto se deve principalmente ao custo da redução do risco e às incertezas associadas. Algum risco residual deve ser sempre aceito. Em situações de alta incerteza, a aceitação do risco torna-se muito problemática devido a essa incerteza. As áreas de processo do SSE-CMM incluem atividades que garantem que a organização analisa ameaças, vulnerabilidades, impactos, e risco associados.

3.1.1.2 Engenharia

Engenharia de segurança é um processo que passa pelo conceito de projeto, implementação, teste, entrega, operação, manutenção, e descontinuidade. Através do processo, engenheiros de segurança devem trabalhar junto com outros membros da equipe de engenharia de sistema.

O SSE-CMM enfatiza que os engenheiros de segurança fazem parte de uma equipe maior e precisam coordenar suas atividades com engenheiros de outras disciplinas. Isto ajuda a garantir que segurança seja parte integrante de processos maiores e não uma atividade distinta e separada. A partir destas informações e de outras sobre requisitos de sistemas e políticas, engenheiros de segurança trabalham com o cliente para identificar necessidades de segurança. Uma vez que as necessidades sejam identificadas, esses engenheiros identificam e traçam requisitos específicos.

O processo de criar soluções para problemas de segurança geralmente envolve identificar possíveis alternativas e então avaliar as alternativas para determinar qual é a mais promissora. A dificuldade em integrar esta atividade com o resto do processo de engenharia é que as soluções não podem ser selecionadas apenas levando-se em consideração a segurança. Há uma ampla variedade de

outras considerações, incluindo custo, desempenho, riscos técnicos e facilidade de uso. As análises produzidas formam uma base significativa para esforços de confiança.

Posteriormente, o engenheiro de segurança é solicitado para garantir que produtos e sistemas estão configurados apropriadamente em relação aos riscos percebidos, assegurando que novos riscos não tornem insegura a operação do sistema.

3.1.1.3 Garantia

A garantia é definida como o nível de confiança pela qual as necessidades de segurança serão satisfeitas (ABRAMS, 1994). O SSE-CMM contribui em relação à garantia na repetição dos resultados de um processo de engenharia de segurança. A base para essa garantia parte do fato de que uma organização madura é mais capaz de repetir bons resultados que uma organização imatura.

A garantia pode ser vista como a confiança de que as proteções funcionarão devidamente. Essa confiança deriva das propriedades de corretude e eficácia. Corretude é a propriedade que as proteções, como projetadas, implementam os requisitos. Eficácia é a propriedade que as proteções fornecem segurança adequada para satisfazer as necessidades de segurança do cliente.

Dessa forma, adicionar quaisquer controles ou proteções para os riscos relacionados com segurança não implica em garantia. Na verdade, a garantia provê a confiança que os controles implementados reduzirão o risco.

A garantia é apoiada por evidências, na forma de documentação, elaboradas durante o curso normal das atividades de engenharia de segurança.

3.1.2 A Arquitetura do SSE-CMM

A arquitetura do SSE-CMM separa características básicas do processo de engenharia de segurança de suas características de gerenciamento e de institucionalização. Para garantir esta separação, o modelo possui duas dimensões: *domínio* e *capacidade*.

A dimensão *domínio* consiste de todas as práticas que coletivamente definem engenharia de segurança. Essas práticas são chamadas de *práticas base*.

A dimensão *capacidade* representa práticas que indicam a gerencia de processo e capacidade de institucionalização. Essas práticas são chamadas de *práticas genéricas*, pois são aplicadas através de um amplo conjunto de domínios, e representam atividades que devem ser realizadas como parte das práticas base.

Relacionando práticas base com práticas genéricas, fornece-se um meio para se verificar a capacidade de uma organização em realizar uma atividade em particular.

O SSE-CMM possui 5 níveis de maturidade, organizados em 22 áreas de processo, que contêm 129 práticas base. Um número de 61 práticas base, organizadas em 11 áreas de processo, cobrem as mais importantes áreas da engenharia de segurança. As 68 práticas base restantes, organizadas em 11 áreas de processo, tratam dos domínios do projeto e organização.

3.1.3 Níveis de Capacidade

Aspectos comuns e níveis de capacidade são importantes tanto para melhorar uma avaliação quanto para melhorar a capacidade do processo da organização. No caso de uma avaliação onde uma organização tem alguns, mas não todos os aspectos comuns implementados em um nível de capacidade particular e para uma área de processo particular, a organização estaria operando no nível completo de capacidade mais baixo para aquele processo. Por exemplo, uma organização que executa quase todas as práticas genéricas do nível 2 para alguma área de processo deveria receber um critério de maturidade de nível 1.

No caso de melhoria, organizar as práticas base em níveis de capacidade provê uma organização com um plano de melhoria caso se deseje melhorar sua capacidade para uma área de processo específica. Por essas razões, as práticas base no SSE-CMM são agrupadas em aspectos comuns os quais são ordenados por níveis de capacidade.

Uma avaliação pode ser realizada para determinar os níveis de capacidade para cada uma das áreas do processo utilizadas na organização, que poderá usar esta informação como um meio de melhorar seus processos, levando em conta seus objetivos de negócio.

O SSE-CMM trabalha com 5 níveis de capacidade (Tabela 3.1).

Tabela 3.1: Níveis de capacidade do SSE-CMM (2003)

Nível	Princípio	Descrição
1 Informalmente Realizado	Fazer antes aquilo que se irá gerenciar.	Define se uma organização ou um projeto realiza um processo que incorpora as práticas base.
2 Planejado e Acompanhado	Entender o que está acontecendo no projeto antes de definir processos amplos da organização.	Foca nos problemas de definição, planejamento e desempenho em nível de projeto.
3 Bem Definido	Usar o melhor que se aprendeu de projetos passados, para criar processos amplos da organização.	Conduz disciplinadamente os processos definidos em nível de organização.
4 Quantitativamente Controlado	Não se pode medir algo até que se saiba o que seja este 'algo'. Gerenciar com medição é apenas significativo, quando se está medindo as coisas certas.	Relaciona medidas com os objetivos de negócio da organização. É essencial que se comece a coletar medidas básicas do projeto desde cedo, embora isto não seja esperado nas organizações até que níveis mais altos tenham sido alcançados.
5	A cultura de melhoria contínua	Recebe influência de todas as melhorias

Melhoria Contínua	requer a fundação de práticas gerenciáveis, processos definidos e metas mensuráveis.	realizadas nos níveis anteriores. Enfatiza que as mudanças culturais sustentarão os ganhos realizados.

3.1.4 Áreas de Processo

Cada área de processo (AP) tem um conjunto de objetivos que representam o estado esperado de uma organização que está realizando com sucesso essa área. Uma organização que realiza as práticas base da área de processo deveria também atingir seus objetivos.

As 11 áreas de processo, que cobrem as mais importantes áreas da engenharia de segurança de sistema do SSE-CMM, são as seguintes:

- AP01 – Administrar Controles de Segurança.
- AP02 – Avaliar Impacto.
- AP03 – Avaliar Riscos de Segurança.
- AP04 – Avaliar Ameaça.
- AP05 – Avaliar Vulnerabilidade.
- AP06 – Construir Argumento de Garantia.
- AP07 – Coordenar Segurança.
- AP08 – Monitorar Postura de Segurança.
- AP09 – Prover Informação de Segurança.
- AP10 – Especificar Necessidades de Segurança.
- AP11 – Verificar e Validar Segurança.

O SSE-CMM também inclui 11 áreas de processo relacionadas às práticas organizacionais e de projeto. Essas áreas de projeto foram adaptadas do SE-CMM:

- AP12 – Garantir Qualidade.
- AP13 – Gerenciar Configuração.
- AP14 – Gerenciar Riscos do Projeto.
- AP15 – Monitorar e Controlar Esforço Técnico.
- AP16 – Planejar Esforço Técnico.
- AP17 – Definir Processo de Engenharia de Sistema da Organização.
- AP18 – Melhorar Processo de Engenharia de Sistema da Organização.
- AP19 – Gerenciar Linha de Evolução do Produto.
- AP20 – Gerenciar Ambiente de Apoio à Engenharia de Sistema.
- AP21 – Fornecer Habilidades e Conhecimento Contínuos.
- AP22 – Coordenar Fornecedores.

As áreas de processo AP12 a AP22 não serão tratadas neste trabalho, uma vez que tratam de engenharia de sistema e, por isso, estão fora do escopo desta pesquisa.

3.1.5 Práticas Genéricas

Práticas genéricas são atividades que se aplicam a todas as áreas de processos. Referem-se ao gerenciamento, à medição, e aos aspectos de institucionalização do processo. Em geral, usadas durante a avaliação para determinar a capacidade de uma organização em realizar o processo.

Práticas genéricas são agrupadas em áreas lógicas chamadas de "Aspectos Comuns", que são projetados para descrever mudanças maiores na forma característica da organização em realizar processo de engenharia de segurança. Cada aspecto comum tem uma ou mais práticas genéricas.

O SSE-CMM não sugere requisitos específicos para realizar as práticas genéricas. Uma organização é livre para planejar, acompanhar, definir, controlar, e melhorar seus processos na forma ou seqüência escolhidas. Contudo, devido à dependência entre algumas práticas genéricas, é sugerido que organizações trabalhem nas práticas genéricas de nível mais baixo antes de tentar os níveis mais altos.

3.1.5.1 Nível de Capacidade 1

No nível de capacidade 1 (Executado Informalmente), as práticas base da área de processo são executadas, embora sua performance não possa ser rigorosamente planejada e acompanhada. Os produtos de trabalho da área de processo são provas de sua performance. Esse nível de capacidade engloba os seguintes aspectos comuns, com suas respectivas práticas genéricas (PG):

- Aspecto comum 1.1 - Práticas base são executadas:
  - PG 1.1.1 – Executar o processo.

3.1.5.2 Nível de Capacidade 2

No nível de capacidade 2 (Planejado e Acompanhado), a execução das práticas base na área do processo é planejada e acompanhada. A execução é verificada de acordo com procedimentos específicos. Os produtos de trabalho estão conforme os requisitos e padrões especificados. A medição é usada para acompanhar a execução da área de processo, bem como possibilitar que a organização gerencie suas atividades baseadas na execução atual. Esse nível de capacidade engloba os seguintes aspectos comuns:

- Aspecto comum 2.1 – Execução é planejada:

- PG 2.1.1 – Alocar recursos.
- PG 2.1.2 – Atribuir responsabilidades.
- PG 2.1.3 – Documentar o processo.
- PG 2.1.4 – Disponibilizar ferramentas.
- PG 2.1.5 – Garantir treinamento.
- PG 2.1.6 – Planejar o processo.

- Aspecto comum 2.2 – A execução é disciplinada:
  - PG 2.2.1 – Usar planos, padrões, e procedimentos.
  - PG 2.2.2 – Realizar gerência de configuração.
- Aspecto comum 2.3 – A execução é verificada:
  - PG 2.3.1 – Verificar aderência do processo.
  - PG 2.3.2 – Auditar produtos de trabalho.
- Aspecto comum 2.4 – A execução é acompanhada:
  - PG 2.4.1 – Acompanhar com medição.
  - PG 2.4.2 – Tomar ações corretivas.

3.1.5.3 Nível de Capacidade 3

No nível de capacidade 3 (Bem Definido), práticas base são executadas de acordo com processo bem definido, usando versões de padrões aprovadas e adaptadas e processos documentados. A distinção em relação ao nível 2 é que o processo é planejado e gerenciado, usando um processo padrão de toda a organização. Esse nível de capacidade engloba os seguintes aspectos comuns:

- Aspecto comum 3.1 – Definir processo padrão:
  - PG 3.1.1 – Padronizar o processo.
  - PG 3.1.2 – Adaptar o processo padrão.
- Aspecto comum 3.2 – Executar o processo definido:
  - PG 3.2.1 – Usar processo bem definido.
  - PG 3.2.2 – Executar revisões de defeito.
  - PG 3.2.3 – Usar dados bem definidos.
- Aspecto comum 3.3 – Coordenar práticas:
  - PG 3.3.1 – Executar coordenação dentro do grupo.
  - PG 3.3.2 – Executar coordenação entre grupos.
  - PG 3.3.3 – Executar coordenação externa.

3.1.5.4 Nível de Capacidade 4

No nível de capacidade 4 (Controlado Quantitativamente), medições detalhadas da performance são coletadas e analisadas. Isto leva a um entendimento quantitativo da capacidade do processo e a uma habilidade melhorada para prever a performance. A performance é objetivamente gerenciada, e a qualidade dos produtos de trabalho é quantitativamente conhecida. A distinção em relação ao nível 3 é que o processo definido é quantitativamente conhecido e controlado. Esse nível de capacidade engloba os seguintes aspectos comuns:

- Aspecto comum 4.1 – Estabelecer metas de qualidade mensuráveis:
  - PG 4.1.1 – Estabelecer metas de qualidade.
- Aspecto comum 4.2 – Objetivamente gerenciar execução.
  - PG 4.2.1 – Determinar a capacidade do processo.
  - PG 4.2.2 – Usar a capacidade do processo.

3.1.5.5 Nível de Capacidade 5

No nível de capacidade 5 (Melhorar Continuamente), metas quantitativas de performance para eficácia e eficiência do processo são estabelecidas baseadas nos objetivos de negócio da organização. Melhorias contínuas do processo em relação a essas metas são definidas por respostas quantitativas da execução dos processos definidos (*feedback*) e da condução de idéias inovadoras e tecnologias. A distinção entre o nível 4 é que o processo definido e o processo padrão passam por refinamento e melhoria contínuos em termos de um entendimento quantitativo do impacto das mudanças nesses processos. Esse nível de capacidade engloba os seguintes aspectos comuns:

- Aspecto comum 5.1 – Melhorar a capacidade organizacional:
  - PG 5.1.1 – Estabelecer metas de eficácia do processo.
  - PG 5.1.2 – Continuamente melhorar o processo padrão.
- Aspecto comum 5.2 – Melhorar a eficácia do processo.
  - PG 5.2.1 – Realizar análise de causa.
  - PG 5.2.2 – Eliminar causas de defeito.
  - PG 5.2.3 – Continuamente melhorar o processo definido.

3.1.6 Práticas Base

As práticas base são necessárias para prover um contexto e apoio para as áreas de processo da engenharia de segurança de sistema. Foram coletadas de um amplo conjunto de materiais existentes da prática e do conhecimento adquirido, sendo práticas já testadas.

Identificar práticas base da engenharia de segurança não é trivial pela grande diversidade de nomes de atividades que são essencialmente as mesmas. Algumas dessas atividades ocorrem em diferentes níveis de abstração ou são realizadas por indivíduos desempenhando papéis distintos.

Algumas questões são relevantes em uma boa prática base:

- Aplica-se ao longo do ciclo de vida.
- Não se sobrepõe a outras práticas base.
- Representa a "melhor prática" da comunidade de segurança.
- É aplicável ao ser utilizada em múltiplos métodos e em vários contextos de negócio.
- Não especifica um método ou uma ferramenta em particular.

Uma organização não pode ser considerada como tendo alcançado uma prática base se essa prática foi apenas realizada no início do ciclo de vida. Todavia, o SSE-CMM identifica o conjunto de práticas que são essenciais para uma boa prática de engenharia de segurança. A seguir, são apresentadas as práticas básicas do SSE-CMM por área de processo.

3.1.6.1 AP01: Gerenciar Controles de Segurança

O propósito desta área de processo é garantir que a segurança para o sistema, que foi integrada ao projeto, seja conseguida pelo sistema resultante em seu estado operacional.

Essa área de processo tem a seguinte meta:

- Configurar e utilizar apropriadamente controles de segurança.

Essa área de processo engloba as seguintes práticas base (PB):

- PB 01.01 – Estabelecer responsabilidades de segurança.
- PB 01.02 – Gerenciar configuração de segurança.
- PB 01.03 – Gerenciar percepção, treinamento e programas de educação de segurança.
- PB 01.04 – Gerenciar serviços de segurança e mecanismos de controle.

Essa área de processo trata das atividades necessárias para administrar e manter os mecanismos de controle de segurança para o ambiente de desenvolvimento e um sistema operacional. Além disso, essa área de processo ajuda a garantir que o nível de segurança não se deteriore. A gerência dos controles para uma nova facilidade deveria se integrar com controles de facilidades existentes.

3.1.6.2 AP02: Avaliar Impacto

O propósito desta área de processo é identificar impactos que sejam relevantes com respeito ao sistema e avaliar a possibilidade da ocorrência desses impactos. Impactos podem

ser tangíveis, tal como a perda da receita ou penalidades financeiras, ou intangíveis, tal como perda de reputação.

Essa área de processo tem a seguinte meta:

- Identificar e caracterizar os impactos de segurança dos riscos do sistema.

Essa área de processo engloba as seguintes práticas base:

- PB 02.01 – Priorizar capacidades.
- PB 02.02 – Identificar ativos do sistema.
- PB 02.03 – Selecionar métrica(s) de impacto.
- PB 02.04 – Identificar relacionamento de métricas.
- PB 02.05 – Identificar e caracterizar impactos.
- PB 02.06 – Monitorar impactos.

Impacto é a conseqüência de um incidente não desejado, causado deliberada ou acidentalmente, o qual afeta os ativos organizacionais. As conseqüências podem ser a destruição de certos ativos, dano no sistema, e perda de confidencialidade, integridade, disponibilidade, responsabilização (*accountability*), autenticidade ou confiabilidade.

Possíveis conseqüências indiretas incluem perdas financeiras, e a perda de mercado ou da imagem da companhia. A medição dos impactos permite um balanço a ser realizado entre os resultados de um incidente não desejado e o custo das proteções contra um incidente não desejado.

A freqüência da ocorrência de um incidente não desejado necessita ser levada em conta. Isto é particularmente importante, quando a quantidade de dano causado por cada ocorrência é baixa, mas onde o efeito agregado de muitos incidentes pode ser prejudicial. A avaliação dos impactos é um elemento importante na avaliação dos riscos e a seleção de proteções.

A informação do impacto produzida por essa área de processo afeta a AP03, AP04 e AP05. Enquanto as atividades envolvidas com agrupar a ameaça, a vulnerabilidade e a informação do impacto estiverem agrupadas em áreas de processo separadas, elas são inter dependentes. O objetivo é encontrar possíveis combinações de ameaças, vulnerabilidades, e impactos, que sejam julgadas suficientemente arriscadas para justificar um conjunto de ações. Todavia, a busca por impactos deveria ser guiada até um certo ponto, pela existência de ameaças e vulnerabilidades correspondentes.

Uma vez que impactos estão sujeitos a mudanças, estes devem ser periodicamente monitorados, para garantir que o entendimento gerado por essa área de processo seja mantido atualizado.

3.1.6.3 AP03: Avaliar Risco de Segurança

O propósito desta área de processo é identificar os riscos de segurança envolvidos com a confiança do sistema em um ambiente definido. Essa área de processo foca no descobrimento desses riscos baseados em um entendimento estabelecido de como capacidades e ativos são vulneráveis a ameaças. Especificamente, esta atividade envolve identificar e avaliar a possibilidade da ocorrência das exposições. "Exposição" refere-se a uma combinação de ameaças, vulnerabilidades e impactos, que poderia causar um dano significativo. Essa atividade pode ser realizada a qualquer momento durante o ciclo de vida do sistema, para apoiar decisões relacionadas com o desenvolvimento, a manutenção, ou a operação do sistema em um ambiente conhecido.

Essa área de processo tem as seguintes metas:

- Atingir o entendimento dos riscos de segurança associado com a operação do sistema em um ambiente definido.
- Priorizar riscos de acordo com uma metodologia definida.

Essa área de processo engloba as seguintes práticas base:

- PB 03.01 – Selecionar métodos de análise de risco.
- PB 03.02 – Identificar exposição.
- PB 03.03 – Avaliar risco de exposição.
- PB 03.04 – Avaliar total incerteza.
- PB 03.05 – Priorizar riscos.
- PB 03.06 – Monitorar riscos e suas características.

Risco de segurança é a probabilidade de o impacto de um incidente não desejado se concretizar. Um risco de segurança lida com proteção contra impactos nos ativos e capacidades do sistema.

Estimativas de risco sempre incluem um fator de incerteza o qual variará dependendo de uma situação em particular. Isto significa que a probabilidade pode apenas ser prevista dentro de certos limites. Além disso, um impacto analisado para um risco em particular também tem associada uma incerteza, uma vez que o incidente não desejado pode não se produzir como esperado. Assim, a maioria dos fatores tem incerteza na acurácia das previsões associadas a eles. Em muitos casos essas incertezas podem ser grandes, tornando o planejamento e a justificativa de segurança muito difícil.

Qualquer ação que possa vir a reduzir a incerteza associada com uma situação em particular é de considerável importância. Por esta razão, a certeza é importante, pois indiretamente reduz o risco do sistema. O objetivo é encontrar combinações de ameaça, vulnerabilidade, e impacto que sejam consideradas suficientemente arriscadas para justificar uma ação.

Considerando-se que ambientes de risco estejam sujeitos a mudanças, eles devem ser periodicamente monitorados para garantir que o entendimento do risco gerado pela área de processo seja mantido atualizado.

3.1.6.4 AP04: Avaliar Ameaça

O propósito desta área de processo é identificar ameaças de segurança e suas propriedades e características.

Essa área de processo tem a seguinte meta:

- Identificar e caracterizar ameaças à segurança do sistema.

Essa área de processo engloba as seguintes práticas base:

- PB 04.01 – Identificar ameaças causadas pela natureza.
- PB 04.02 – Identificar ameaças causadas por pessoas.
- PB 04.03 – Identificar unidades de medida da ameaça.
- PB 04.04 – Avaliar a capacidade do agente da ameaça.
- PB 04.05 – Avaliar a probabilidade da ameaça.
- PB 04.06 – Monitorar ameaças e suas características.

Muitos métodos e metodologias podem ser usados para realizar uma avaliação de ameaças. Uma consideração importante para determinar que metodologia usar é como ela comunicará e trabalhará com as metodologias usadas em outras partes do processo de avaliação de risco.

A informação da ameaça produzida por essa área de processo pode ser usada em AP03, AP05 e AP02. Enquanto as atividades envolvidas com o conjunto de informações de ameaça, de vulnerabilidade, e de impacto forem agrupadas em áreas de processo separadas, elas são interdependentes. O objetivo é encontrar combinações de ameaça, vulnerabilidade, e impacto que sejam consideradas arriscadas para justificar a ação. Portanto, a busca por ameaças deveria ser guiada até um certo ponto pela existência de vulnerabilidades e impactos correspondentes.

Uma vez que as ameaças estão sujeitas a mudanças, elas devem ser periodicamente monitoradas para garantir que o entendimento gerado por essa área de processo seja mantido atualizado.

3.1.6.5 AP05: Avaliar Vulnerabilidade

O propósito desta área de processo é identificar e caracterizar vulnerabilidades de segurança do sistema. Esta área de processo inclui analisar os ativos do sistema, definir vulnerabilidades específicas, e fornecer uma avaliação geral de vulnerabilidade do sistema. Os

termos associados com risco de segurança e avaliação de vulnerabilidade são usados diferentemente em muitos contextos.

Para os propósitos do SSE-CMM, "vulnerabilidade" refere-se a um aspecto de um sistema, que pode ser explorado para outros propósitos em detrimento daqueles originalmente pensados, bem como fraquezas, buracos de segurança, ou falhas de implementação dentro do sistema, que sejam possíveis de serem atacados pela ameaça. Essas vulnerabilidades são independentes de qualquer instância de ameaça ou ataque em particular. Este conjunto de atividades pode ser executado a qualquer momento durante um ciclo de vida do sistema para apoiar a decisão de desenvolver, manter, ou operar o sistema dentro do ambiente conhecido.

Essa área de processo tem a seguinte meta:

- Conseguir um entendimento das vulnerabilidades da segurança do sistema dentro de um ambiente definido.

Essa área de processo engloba as seguintes práticas base:

- PB 05.01 – Selecionar método de análise de vulnerabilidades.
- PB 05.02 – Identificar vulnerabilidades.
- PB 05.03 – Agrupar dados das vulnerabilidades.
- PB 05.04 – Produzir vulnerabilidades do sistema.
- PB 05.05 – Monitorar vulnerabilidades e suas características.

Descobrir vulnerabilidades do sistema por ferramentas e técnicas ativas é outro método que complementa, mas não substitui outras técnicas de análise de vulnerabilidades. Essas técnicas ativas podem ser vistas como uma forma especializada de análise de vulnerabilidade. Esse tipo de análise pode ser útil, quando se tenta validar vulnerabilidades de segurança de um sistema após sua atualização ou para identificar vulnerabilidades de segurança, quando dois ou mais sistemas são interconectados.

Análise de vulnerabilidade ativa é necessária em alguns casos para validar a postura de segurança do sistema e para elevar a percepção e entendimento das vulnerabilidades de segurança existentes. Às vezes, referida como teste de invasão, é um processo no qual engenheiros de segurança tentam evitar aspectos de insegurança do sistema. Os engenheiros de segurança tipicamente trabalham sobre as mesmas restrições aplicadas a usuários comuns, mas pode ser assumido a usar toda a documentação do projeto e implementação. O processo de atacar a segurança não é exaustivo e é restringido por tempo e recursos financeiros.

A informação de vulnerabilidade produzida por essa área de processo é abordada em AP03, AP04 e AP02. Enquanto as atividades envolvidas com o conjunto de informações de

ameaça, de vulnerabilidade e de impacto forem agrupadas em áreas de processo separadas, elas são interdependentes. O objetivo é encontrar combinações de ameaça, vulnerabilidade e impacto que sejam consideradas arriscadas, para justificar possíveis ações. Portanto, a busca por vulnerabilidades deveria ser guiada até um certo ponto pela existência de ameaças e impactos correspondentes.

Uma vez que as vulnerabilidades estejam sujeitas a mudanças, elas devem ser periodicamente monitoradas, para garantir que o entendimento gerado por essa área de processo seja mantida atualizada.

3.1.6.6 AP06: Construir Argumento de Garantia

O propósito desta área de processo é expressar que as necessidades de segurança do cliente sejam satisfeitas. Um argumento de garantia é um conjunto de objetivos de garantia que sejam apoiados pela combinação de evidências, que podem ser derivadas de diversas fontes e níveis de abstração.

Esse processo inclui identificar e definir requisitos relacionados com a garantia, produção de evidências e atividades de análise, e atividades de evidência adicionais necessárias, para dar o suporte aos requisitos de garantia. Adicionalmente, a evidência gerada por essas atividades deve ser agrupada, organizada e preparada para apresentação.

Essa área de processo tem a seguinte meta:

- Os produtos de trabalho e processos fornecem a evidência de que as necessidades de segurança do cliente estão sendo atingidas.

Essa área de processo engloba as seguintes práticas base:

- PB 06.01 – Identificar objetivos de garantia da segurança.
- PB 06.02 – Definir estratégia de garantia.
- PB 06.03 – Controlar evidência de garantia.
- PB 06.04 – Analisar evidência.
- PB 06.05 – Fornecer argumento de garantia.

As atividades envolvidas na construção de argumentos de garantia incluem: gerenciar a identificação, o planejamento, a organização e a apresentação de evidências de garantia da segurança.

3.1.6.7 AP07: Coordenar Segurança

O propósito desta área de processo é garantir que todas as partes estejam cientes e envolvidas com as atividades de engenharia de segurança. Esta atividade é crítica, uma vez

que engenharia de segurança precisa interagir com outras engenharias. A coordenação envolve manter comunicação entre toda equipe do projeto e grupos externos. Vários mecanismos podem ser usados para coordenar e comunicar as decisões e recomendações da engenharia de segurança entre estas partes, incluindo memorandos, documentos, mensagem eletrônica, reuniões, e grupos de trabalho.

Essa área de processo tem as seguintes metas:

- Todos os membros da equipe do projeto devem estar cientes e envolvidos com as atividades de engenharia de segurança a ponto de realizar suas funções.
- As decisões e recomendações relacionadas com segurança devem ser comunicadas e coordenadas.

Essa área de processo engloba as seguintes práticas base:

- PB 07.01 – Definir relacionamentos e objetivos de coordenação da engenharia de segurança.
- PB 07.02 – Identificar mecanismos de coordenação para engenharia de segurança.
- PB 07.03 – Facilitar a coordenação da engenharia de segurança.
- PB 07.04 – Utilizar os mecanismos identificados para coordenar decisões e recomendações relacionadas com segurança.

Esta área de processo garante que segurança seja uma parte integrante de todo o esforço de engenharia. A engenharia de segurança deveria envolver as principais equipes de projeto e grupos de trabalho. É importante que a engenharia de segurança estabeleça relacionamentos com outras equipes de engenharia, já nas fases iniciais do ciclo de vida, quando decisões críticas de projeto são tomadas. Essa área de processo pode ser igualmente aplicada tanto em organizações de desenvolvimento quanto naquelas operacionais.

3.1.6.8 AP08: Monitorar Postura de Segurança

O propósito desta área de processo é garantir que todas as deficiências ou erros que poderiam conduzir a uma "brecha" de segurança sejam identificados e reportados. Os ambientes externos e internos são monitorados por todos os fatores que podem ter um impacto na segurança do sistema.

Essa área de processo tem as seguintes metas:

- Os eventos externos e internos relacionados com segurança são detectados e acompanhados.
- Incidentes são tratados de acordo com a política estabelecida.

- Mudanças na postura de segurança operacional são identificadas e manipuladas de acordo com objetivos de segurança.

Essa área de processo engloba as seguintes práticas base:

- PB 08.01 – Analisar registros de evento para determinar a causa do evento, sua ocorrência, e probabilidade de eventos futuros.
- PB 08.02 – Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, e ambiente.
- PB 08.03 – Identificar incidentes de segurança relevantes.
- PB 08.04 – Monitorar a eficácia funcional e a performance de proteções de segurança.
- PB 08.05 – Revisar a postura de segurança do sistema para identificar mudanças necessárias.
- PB 08.06 – Gerenciar a reposta dos incidentes de segurança relevantes.
- PB 08.07 – Garantir que os artefatos relacionados com monitoramento de segurança são adequadamente protegidos.

Uma postura de segurança indica o preparo do sistema e de seu ambiente para manipular ameaças atuais, e vulnerabilidades e qualquer impacto ao sistema e seus ativos. Essa área de processo envolve as atividades da área de processo AP05 e AP03. Os dados agrupados sobre o ambiente externo e interno são analisados em seu próprio contexto e em relação a outros dados, que podem resultar dos eventos ocorridos antes, em paralelo com ou após um evento em questão. Essa área trata tanto do ambiente alvo do sistema quanto do ambiente no qual o sistema é desenvolvido. Qualquer sistema em particular tem que funcionar em conjunção com sistemas existentes, que podem afetar sua segurança geral. Assim, esses sistemas existentes deveriam ser incluídos no monitoramento.

3.1.6.9 AP09: Fornecer Informação de Segurança

O propósito desta área de processo é prover aos arquitetos de sistema, projetistas, implementadores, ou usuários a informação de segurança de que eles necessitam. Essa informação inclui arquitetura segura, projeto, ou alternativas de implementação e orientação de segurança. A informação é desenvolvida, analisada, fornecida, e coordenada com os membros apropriados da organização, baseada nas necessidades de segurança identificadas em AP10 (Especificar Necessidades de Segurança).

Essa área de processo tem as seguintes metas:

- Todos os problemas do sistema são revisados em relação a implicações de segurança e são resolvidos de acordo com objetivos de segurança.

- Todos os membros da equipe de projeto compreendem segurança e, por conseguinte, podem realizar suas funções.
- A solução reflete a informação de segurança fornecida.

Essa área de processo engloba as seguintes práticas base:

- PB 09.01 – Trabalhar com projetistas, desenvolvedores, e usuários para garantir que as partes tenham um entendimento comum das necessidades de informação de segurança
- PB 09.02 – Determinar ponderações e restrições de segurança necessárias para realizar escolhas de segurança de maneira consciente.
- PB 09.03 – Identificar soluções alternativas para problemas relacionados com engenharia de segurança.
- PB 09.04 – Analisar e priorizar alternativas de engenharia usando ponderações e restrições de segurança.
- PB 09.05 – Fornecer orientação de segurança a outros grupos de engenharia.
- PB 09.06 – Fornecer orientação de segurança aos usuários e administradores de sistema operacional.

Esta área de processo fornece informação de segurança para apoiar o projeto do sistema e atividades de implementação. Seu foco é como a segurança torna-se parte integrante do desenvolvimento do sistema e não um fim em si mesmo. Cada uma das práticas base utiliza informações de toda a organização, produz resultados específicos de segurança, e comunica esses resultados de volta para toda organização. Os processos identificados são aplicáveis no desenvolvimento de novas facilidades ou na operação e manutenção de facilidades existentes.

Essa área de processo cobre informação de segurança para desenvolvimento (projetistas, e implementadores) e para operação (usuários e administradores). Além disso, combinar atividades de projeto e implementação de segurança em uma única área de processo, isto enfatiza que essas atividades são muito semelhantes; porém, estão em diferentes níveis de abstração. As soluções alternativas variam em escopo, de completas arquiteturas de sistema a componentes individuais. Alguns aspectos de requisitos de segurança têm impacto no ambiente onde o sistema é desenvolvido em vez do próprio sistema.

Todas as práticas base dentro dessa área de processo podem ser iterativas e todas ocorrem em múltiplos pontos através do ciclo de vida do sistema.

3.1.6.10 AP10: Especificar Necessidades de Segurança

O propósito desta área de processo é explicitamente identificar as necessidades relacionadas com segurança para o sistema. Envolve definir as bases para segurança no sistema de modo a satisfazer todos os requisitos legais e organizacionais para segurança. Essas necessidades baseiam-se no contexto de segurança operacional do sistema, no atual ambiente de segurança e sistema da organização. Um conjunto de requisitos de segurança é definido para o sistema, tornando-se uma *baseline* de segurança para o sistema sob aprovação.

Essa área de processo tem a seguinte meta:

- Entendimento comum das necessidades de segurança é alcançado entre todas as partes, incluindo o cliente.

Essa área de processo engloba as seguintes práticas base:

- PB 10.01 – Obter compreensão das necessidades de segurança do cliente.
- PB 10.02 – Identificar leis, políticas, padrões, restrições e influências externas que se relacionam com o sistema.
- PB 10.03 – Identificar o propósito do sistema para determinar o contexto de segurança.
- PB 10.04 – Capturar uma visão de alto nível orientada à segurança da operação do sistema.
- PB 10.05 – Capturar metas de alto nível, que definem a segurança do sistema.
- PB 10.06 – Definir um conjunto consistente de declarações que estabeleçam a proteção a ser implementada no sistema.
- PB 10.07 – Obter acordo comprovando que os requisitos específicos de segurança envolvem as necessidades do cliente.

Essa área de processo cobre as atividades, que definem todos os aspectos de segurança no sistema de informação (e.g. físico, funcional, procedimental). As práticas base tratam de como as necessidades de segurança são identificadas e refinadas em uma *baseline* coerente de requisitos de segurança que é utilizada em projeto, desenvolvimento, verificação, operação, e manutenção do sistema. Em muitos casos, é necessário levar em conta o ambiente existente e as necessidades de segurança associadas. A informação obtida e produzida por essa área de processo é coletada, refinada, utilizada, e atualizada através do projeto (particularmente na AP09), de modo a garantir que as necessidades do cliente sejam atendidas e tratadas.

3.1.6.11 AP11: Verificar e Validar Segurança

O propósito desta área de processo é garantir soluções verificadas e validadas com respeito à segurança. Soluções são verificadas em relação aos requisitos de segurança, à arquitetura e ao projeto usando observação, demonstração, análise e teste. Soluções são validadas em relação às necessidades de segurança operacional do cliente.

Essa área de processo tem as seguintes metas:

- Soluções envolvem os requisitos de segurança.
- Soluções envolvem as necessidades de segurança operacional do cliente.

Essa área de processo engloba as seguintes práticas base:

- PB 11.01 – Identificar a solução a ser verificada e validada.
- PB 11.02 – Definir a abordagem e nível de rigor para verificar e validar cada solução.
- PB 11.03 – Verificar que a solução implementa os requisitos associados com o nível anterior de abstração.
- PB 11.04 – Validar a solução mostrando que ela satisfaz as necessidades associadas com o nível anterior de abstração, e as necessidades de segurança operacional do cliente.
- PB 11.05 – Capturar os resultados de verificação e validação para outros grupos de engenharia.

Esta área de processo é uma parte importante da verificação e validação do sistema e ocorre em todos os níveis de abstração. Soluções incluem tudo desde conceitos operacionais a arquiteturas e implementações, continuando em todo o sistema de informação, incluindo ambiente e procedimentos.

No interesse de obter resultados objetivos, o grupo de verificação e validação deveria ser um grupo diferente dos grupos de engenharia. Contudo, esse grupo pode trabalhar ao lado dos grupos de engenharia. Os resultados da verificação e validação podem ser repassados a todos os grupos de engenharia em qualquer momento durante o ciclo de vida da solução. Verificação e validação são normalmente associadas com os conceitos de corretude e eficácia.

3.1.7. Passo a Passo para Utilizar o SSE-CMM

A seguir será apresentado um *checklist*, para a utilização do SSE-CMM:

i. Selecione uma área de processo que possa se adequar na missão de sua organização.

ii. Revise a descrição, as metas e as práticas base da área de processo selecionada.

iii. Verifique se estão sendo realizadas as práticas base para a área de processo selecionada. Não é mandatória a realização de todas as práticas dessa área.

iv. Verifique se a organização está atingindo as metas para a área de processo selecionada.

v. Selecione o *aspecto comum* 2.1, "Execução é Planejada", e revise sua descrição e suas práticas genéricas.

vi. Verifique se sua organização está planejando a performance da área de processo selecionada. Isto deve levar em consideração as práticas genéricas descritas no *aspecto comum* 2.1.

vii. Repita os passos de *iv* a *vi* para cada um dos outros *aspectos comuns*.

viii. Repita os passos de *i* a *vii* para outras áreas de processo.

Os passos do *checklist* podem auxiliar na utilização do SSE-CMM. Contudo, uma avaliação completa com auditores independentes e múltiplos participantes poderá conduzir a uma visão mais acurada de utilização do SSE-CMM.

3.2 ISO/IEC 27002

Para a ISO/IEC 27002 (2005) **–** *Código de prática para a gestão da segurança da informação* – a informação é um ativo que agrega valor para a organização e, conseqüentemente, necessita ser adequadamente protegida. A segurança da informação protege a informação de diversos tipos de ameaças para garantir a continuidade dos negócios, minimizar os danos aos negócios e maximizar o retorno do investimento e as oportunidades de negócio.

Anteriormente, esta norma se chamava ISO 17799 e, em julho de 2007, teve a nova numeração para se adequar ao grupo de normas ISO 27000 sobre técnicas de segurança.

Esta norma objetiva preservar a confidencialidade, integridade e disponibilidade das informações. Isso é conseguido com a implementação de controles, podendo ser políticas, práticas ou processos. Esses controles garantem que objetivos estabelecidos para a segurança serão satisfeitos.

Segundo a ISO/IEC 27002 (2005), é essencial para uma organização identificar seus requisitos de segurança. Isso pode ser conseguido, primeiramente, pela avaliação de risco dos ativos através da análise de ameaças, vulnerabilidades, probabilidade da ameaça se concretizar, bem como o impacto dessa ameaça na organização. Outra fonte é identificar legislação, normas, regulamentos, contratos, entre outros a que a organização deva atender. Finalmente, as estruturas a serem desenvolvidas para apoiar as operações organizacionais, ou

seja, seus princípios, objetivos e requisitos de processamento da informação, também identificam os requisitos de segurança.

Requisitos de segurança e controles estão intrinsecamente relacionados à medida que, para cada requisito de segurança, um ou mais controles devam ser implementados para reduzir os riscos identificados.

A ISO/IEC 27002 é também uma fonte de controles, que podem ser selecionados e utilizados. Não obstante, novos controles podem ser criados para também atender a necessidades levantadas. No entanto, salientamos que alguns controles não se aplicam a todos os sistemas de informação. Além disso, é preciso verificar as condições de custo para implementar um determinado controle, isto é, o custo não deve ultrapassar o valor das perdas causadas pelo risco. Convém que aspectos não financeiros, como danos à imagem da organização, sejam também considerados.

Apresentamos abaixo um resumo dos principais fatores críticos de sucesso, para que a segurança da informação seja implementada (ISO/IEC 27002, 2005)

- elaborar política de segurança, objetivos e atividades, que reflitam os objetivos de negócio;
- ter um bom entendimento dos requisitos de segurança, avaliação de risco e da gestão de risco;
- distribuir diretrizes e normas sobre a política de segurança da informação a todos envolvidos com a organização;
- proporcionar conscientização, educação e treinamento adequados;
- utilizar um sistema de medição para avaliar o desempenho da gestão de segurança da informação.

3.2.1 Controles da ISO/IEC 27002

Apresentamos a seguir apenas os controles da ISO/IEC 27002, organizados em diversos itens desta norma, e que foram utilizados no contexto deste trabalho. Os demais controles não foram considerados porque não influenciavam a proposta do trabalho.

3.2.1.1 Analisando / Avaliando os Riscos de Segurança da Informação

Este controle (Item 4.1 da norma) prega que as análises/avaliações de riscos identifiquem, quantifiquem, e priorizem, periodicamente, os riscos com base em critérios para aceitação dos riscos e dos objetivos relevantes para organização. Os resultados das análises devem orientar e determinar as ações de gestão adequadas, as prioridades para o

gerenciamento dos riscos, e a implementação dos controles selecionados, a fim de se proteger desses riscos.

3.2.1.2 Tratando os Riscos de Segurança da Informação

Este controle (Item 4.2 da norma) diz que, antes de se tratar um risco, a organização deve definir os critérios de aceitabilidade ou não desses riscos. A aceitação do risco implica que o risco é baixo ou que o custo de seu tratamento é economicamente inviável para a organização. Para os riscos que serão tratados, controles devem ser selecionados e implementados para atender aos requisitos identificados pela análise/avaliação de riscos.

3.2.1.3 Infra-estrutura da Segurança da Informação

Este controle (Item 6.1 da norma) propõe que uma estrutura de gerenciamento seja estabelecida para iniciar e controlar a implementação da segurança da informação dentro da organização. Fóruns, comitês ou grupos apropriados de gerenciamento são estabelecidos para aprovar uma política de segurança da informação, atribuir as funções da segurança e coordenar a implementação da segurança através da organização.

3.2.1.4 Responsabilidade pelos Ativos

Este controle (Item 7.1 da norma) diz que os ativos de informação sejam inventariados e possuam um proprietário que seja responsabilizado pela manutenção apropriada dos controles que mitigam os riscos dos ativos. O inventário desses ativos apóia a garantia de que as proteções estão sendo feitas de maneira eficaz.

3.2.1.5 Classificação da Informação

Este controle (Item 7.2 da norma) propõe que a classificação da informação e seus respectivos controles de proteção levem em consideração as necessidades de compartilhamento, seus requisitos legais, restrição de informações e os respectivos impactos nos negócios.

Da mesma forma, é conveniente que informações e resultados de sistemas, que processam dados classificados, sejam rotulados de acordo com seu valor, sua sensibilidade e sua criticidade para a organização.

A responsabilidade pela definição da classificação de um item de informação, tal como um registro de dado, e a análise crítica periódica desta classificação devem ser do autor ou proprietário responsável pela informação.

3.2.1.6 Procedimentos e Responsabilidades Operacionais

Este controle (Item 10.1 da norma) relata que as modificações nos sistemas e recursos de processamento da informação devem ser controladas. O controle inadequado de modificações nos sistemas é uma causa comum da falha de segurança.

Espera-se que haja formalização dos procedimentos e das responsabilidades, para garantir que exista um controle satisfatório de todas as mudanças de *software*, procedimentos, entre outros. Programas disponibilizados em produção devem passar por controles específicos de modificação, realizando e mantendo uma trilha de auditoria com informações relevantes.

Este controle define a segregação de funções como um método para redução de riscos de mau uso acidental ou deliberado dos sistemas, bem como para evitar fraudes de difícil detecção. É necessário separar as responsabilidades de autorização e a execução de determinados processos, a fim de reduzir ações não autorizadas ou mau uso das informações ou dos processos.

A separação dos ambientes de desenvolvimento, teste e produção é importante para se alcançar a segregação de funções envolvidas. É conveniente que as regras para transferir o *software* do ambiente de desenvolvimento para produção sejam bem definidas e documentadas.

Quando as equipes de desenvolvimento e teste possuem acesso ao ambiente de produção, existe a possibilidade de se introduzir códigos não testados ou não autorizados, ou mesmo alterar os dados reais dos sistemas. Dependendo do sistema, essas ações podem levar a fraudes.

3.2.1.7 Planejamento e Aceitação dos Sistemas

Este controle (Item 10.3 da norma) foca em minimizar o risco de falhas nos sistemas. Entre outras recomendações, afirma-se ser conveniente que os requisitos de capacidade sejam identificados e monitorados e que as projeções de cargas de produção futura sejam feitas de forma a garantir a disponibilidade da capacidade adequada de processamento e armazenamento. Tais projeções devem considerar os requisitos de novos negócios e sistemas e as tendências atuais e projetadas de capacidade de processamento de informação da organização.

Outro fator é o estabelecimento de critérios de aceitação de novos sistemas, atualizações e novas versões, bem como a realização de testes dos sistemas antes da aceitação. É esperado que os gestores garantam que os requisitos e critérios para aceitação de novos sistemas estejam claramente definidos, acordados, documentados e testados.

Em relação a novos projetos de desenvolvimento, convém que usuários sejam consultados em todos os estágios do processo de desenvolvimento, de forma a garantir a eficiência operacional do projeto proposto e sua adequação às necessidades organizacionais. Desta forma, os testes devem garantir que os critérios de aceitação sejam satisfeitos.

3.2.1.8 Monitoramento

Este controle (Item 10.10 da norma) focaliza em monitorar os sistemas para detectar divergências entre a política de controle de acesso e os registros de eventos monitorados, fornecendo evidências no caso de incidentes de segurança.

3.2.1.9 Gerenciamento de Acessos do Usuário

Este controle (Item 11.2 da norma) trata da prevenção de acessos não autorizados aos sistemas de informação. Neste caso, procedimentos formais devem ser estabelecidos para controlar a concessão de direitos de acesso aos sistemas de informação e serviços.

Esses procedimentos devem cobrir todos os estágios do ciclo de vida de acesso de um usuário, do registro inicial de novos usuários até o registro final de exclusão dos usuários, que não mais necessitam ter acesso aos sistemas.

3.2.1.10 Controle de Acesso à Aplicação e à Informação

Este controle (Item 11.6 da norma) focaliza a prevenção de acesso não autorizado à informação contida nos sistemas de informação. Convém que o acesso lógico à aplicação e informação seja restrito a usuários autorizados. Convém que os sistemas de aplicação:

- controlem o acesso dos usuários à informação e às funções dos sistemas de aplicação, de acordo com uma política definida de controle do negócio;
- proporcionem proteção contra acesso não autorizado para qualquer software que seja capaz de sobrepor ou contornar os controles das aplicações ou do sistema;
- não comprometam a segurança de outros sistemas com os quais os recursos de informação sejam compartilhados.

3.2.1.11 Requisitos de Segurança de Sistemas de Informação

Este controle (Item 12.1 da norma) garante que a segurança seja parte integrante dos sistemas de informação. Assim, convém que todos os requisitos de segurança sejam identificados na fase de definição de requisitos de um projeto e justificados, acordados e documentados como parte do estudo do negócio para um sistema de informação.

Convém que os requisitos e controles de segurança reflitam o valor, para o negócio, dos ativos de informação envolvidos e o dano potencial ao negócio, que possa resultar da falha ou ausência de segurança. A estrutura para analisar os requisitos de segurança e identificar os controles que os satisfazem deve estar na avaliação de riscos e no gerenciamento de riscos.

3.2.1.12 Processamento Correto nas Aplicações

Este controle (Item 12.2 da norma) previne a ocorrência de erros, perdas, modificação não autorizada ou uso impróprio de dados do usuário nos sistemas de aplicações.

Convém que controles apropriados e trilhas de auditoria ou registros de atividades sejam previstos para os sistemas de aplicação e incorporados no projeto destes sistemas, incluindo aplicações escritas pelo usuário, para assegurar o correto processamento. Convém também que estes incluam a validação dos dados de entrada, processamento interno e dados de saída. Este controle deve ser determinado com base em requisitos de segurança e na avaliação de riscos.

3.2.1.13 Gestão de Vulnerabilidades Técnicas

Este controle (Item 12.6 da norma) implementa repetidamente uma gestão efetiva e sistemática de vulnerabilidades técnicas, para reduzir riscos resultantes da exploração de vulnerabilidades.

3.2.1.14 Notificação de Fragilidades e Eventos de Segurança da Informação

Este controle (Item 13.1 da norma) prega que um procedimento de notificação formal seja estabelecido, definindo-se a ação a ser tomada ao se receber uma notificação de incidente. Por exemplo: o mau funcionamento de *software* que possa ter impacto na segurança dos ativos organizacionais.

3.2.1.15 Gestão de Incidentes de Segurança da Informação e Melhorias

Este controle (Item 13.2 da norma) define responsabilidades e procedimentos para o adequado tratamento de eventos de segurança da informação, assegurando um enfoque consistente e efetivo a ser aplicado na gestão de incidentes e garantindo sua melhoria.

É necessário existir mecanismos para quantificar e monitorar tipos, quantidades e custos de incidentes e maus funcionamentos. Tal informação pode ser usada para identificar incidentes ou maus funcionamentos recorrentes ou de alto impacto, podendo indicar a necessidade de melhoria ou controles extras para reduzir freqüência, danos e custos de futuras ocorrências ou para ser levado em consideração, quando for realizado o processo de análise crítica da política de segurança.

3.2.1.16 Conformidade com Requisitos Legais

Este controle (Item 15.1 da norma) evita a violação de qualquer lei criminal ou civil, estatutos, regulamentações ou obrigações contratuais e de quaisquer requisitos de segurança.

Convém que estatutos, regulamentações ou cláusulas contratuais relevantes sejam explicitamente definidos e documentados para cada sistema de informação. Os controles e as responsabilidades específicos para atender a estes requisitos devem ser definidos e documentados.

Procedimentos apropriados devem ser implementados para garantir a conformidade com as restrições legais no uso de material de acordo com leis de propriedade intelectual, como as de direitos autorais. A violação do direito autoral pode levar a uma ação legal envolvendo processos criminais.

3.2.1.17 Conformidade com Normas e Políticas de Segurança da Informação e Conformidade Técnica

Este controle (Item 15.2 da norma) promove a garantia da conformidade dos sistemas com as políticas e normas organizacionais de segurança da informação. Convém que a segurança dos sistemas de informação seja analisada criticamente em intervalos regulares e que tais análises críticas sejam executadas com base nas políticas de segurança apropriadas, auditando em conformidade com as normas de segurança implementadas, as plataformas técnicas e os sistemas de informação.

3.3 ISO/IEC 15408

O *Common Criteria* (CC, 2005) tornou-se a norma ISO/IEC 15408 (2005a, 2005b, 2005c) – *Critérios de Avaliação para a Segurança da Tecnologia da Informação* – sendo o critério *defacto* para avaliar a segurança de produtos da Tecnologia da Informação (TI).

Segundo Albuquerque (2002), o *Common Criteria* fornece conjunto de critérios fixos que permitem especificar a segurança de uma aplicação de forma não ambígua a partir de características do ambiente da aplicação e definir formas de garantir a segurança da aplicação para o cliente final.

A norma esclarece que o desenvolvimento seguro de *software* envolve segurança no ambiente de desenvolvimento (espaço físico restrito; existência de ambientes de desenvolvimento e de produção separados; processo de desenvolvimento formalmente definido) e segurança da aplicação desenvolvida (definição de requisitos e controles de segurança).

A ISO/IEC 15408 é formada por três partes:

- ISO/IEC 15408 – Parte 1: contém a introdução e o modelo geral, a descrição do modelo da norma, e terminologias e conceitos.

- ISO/IEC 15408 – Parte 2: contém o catálogo dos requisitos funcionais de segurança organizados em classes, famílias, componentes e elementos.
- ISO/IEC 15408 – Parte 3: contém o catálogo dos requisitos de garantia de segurança organizados em classes, famílias, componentes e elementos, como também a definição dos níveis de garantia de avaliação.

O *Common Criteria* se diferencia da ISO/IEC 15408 por possuir uma metodologia de avaliação da segurança de tecnologia da informação denominada CEM (*Common Criteria Evaluation Methodology*) (CEM, 2006).

O CEM descreve as ações mínimas a serem executadas por um avaliador a fim de conduzir uma avaliação de aderência ao *Common Criteria*, utilizando os critérios e a evidência da avaliação definidos no *Common Criteria*. O certificador do *Common Criteria* se utiliza do CEM para ratificar as ações dos avaliadores.

Segundo Prieto-Díaz (2004), o *Common Criteria* é um conjunto de requisitos reutilizáveis que podem ser agrupados para construir documentos de requisitos funcionais e de garantia. Contudo, o foco geral do *Common Criteria* e da ISO/IEC 15408 é verificar que o documento com os requisitos de um determinado produto de TI está em conformidade com os requisitos definidos no *Common Criteria* ou na ISO/IEC 15408.

3.3.1 Organização dos Requisitos de Segurança

Os requisitos de segurança da ISO/IEC 15408 são dispostos hierarquicamente na:

- ISO/IEC 15408-2: define 11 classes de segurança funcional com 67 famílias de segurança funcional. Mookhey (2005) afirma que essa parte define um vetor de funções ou características de segurança que pode ser usado para avaliação ou durante o desenvolvimento.
- ISO/IEC 15408-3: define 9 classes de garantia de segurança, com 40 famílias de garantia de segurança. Mookhey (2005) afirma que essa parte define um vetor de componentes de garantia que pode ser selecionado durante o processo de avaliação.

De acordo com Prieto-Díaz (2004), uma classe é um conjunto de requisitos de segurança que compartilham um foco comum. Os membros de uma classe formam famílias, que representam uma entidade de nível mais alto, em que arquitetos de sistema e desenvolvedores selecionam os requisitos.

Componentes são um conjunto específico de requisitos de segurança formados por elementos. Elemento é um requisito de segurança indivisível, que pode ser verificado por uma ação

de avaliação. Ação de avaliação é um conjunto de passos ou "unidades de trabalho", que um avaliador deve concluir para avaliar e verificar um requisito de segurança.

A ISO/IEC 15408-3 especifica como os PPs (*Protection Profiles* – Perfis de proteção) e STs (*Security Target* – Alvo de segurança) devem ser avaliados e identifica sete EALs (*Evaluation Assurance Level* – Nível de garantia de avaliação).

EAL é um grupo de componentes de garantia que representa um marco em uma escala pré-definida de garantia. EAL1 é o menor nível de confiança, garantindo apenas que o TOE (*Target of Evaluation* – Alvo da Avaliação) funciona. EAL7 provê o maior nível de confiança e certifica que o projeto do produto (TOE) foi formalmente verificado e exaustivamente testado.

3.3.1.1 Requisitos Funcionais de Segurança

As classes funcionais de segurança (Tabela 3.2) representam as funções mais amplas de segurança necessárias para um produto de TI.

Tabela 3.2: Classes dos requisitos funcionais (MOOKHEY, 2005)

Classe		**Descrição**
Nome	**Sigla**	
Auditoria de segurança	FAU	Lida com a funcionalidade de trilha de auditoria fornecida pelo produto, envolvendo as atividades que serão auditadas, o armazenamento seguro dessas trilhas, e a funcionalidade de analisar e revisar os registros dessas trilhas.
Comunicação	FCO	Trabalha tanto com o não repúdio da informação de origem quanto da informação de destino.
Apoio criptográfico	FCS	Gerencia a geração, a distribuição, o acesso, e a destruição de chaves. Trata do uso da chave na encriptação/decriptação, geração de *hash* ou *checksum*, assinatura digital, entre outras operações. Especifica também o algoritmo criptográfico a ser usado e o tamanho das chaves.
Proteção dos dados do usuário	FDP	Trata da proteção dos dados do usuário que são armazenados e transmitidos pelo TOE. Especifica a criação de uma política de controle de acesso e fluxo de informação ao redor da qual as funções de segurança devem ser projetadas e implementadas.
Identificação e autenticação	FIA	Determina como um usuário é identificado (por nome ou identificador de usuário) e autenticado (senha, *smartcard*, ou sistemas de biometria) pelo sistema. Faz também a atribuição dos atributos corretos de segurança ao usuário autenticado.
Gerência de segurança	FMT	Trabalha com as características internas do produto para a gerência de segurança, tais como listas de controle de acesso, parâmetros de segurança, definição e atribuição de papéis de segurança, etc.
Privacidade	FPR	Protege a identidade do usuário no sistema.
Proteção das funções de segurança do TOE	FPT	Cobre a proteção do TOE em termos de componentes que o constituem. Está relacionada a questões do tipo: • *O produto trabalha em modo de segurança*? • *O produto realiza testes de inicialização*? • *É possível recuperar o produto de falhas, protegendo-se ainda suas informações*? • *A checagem de controle de acesso é executada antes de permitir que sejam realizadas ações restritas*?

Utilização de recursos	FRU	Garante que recursos tais como capacidade de processamento e armazenamento sejam gerenciados para prevenir condições de falta de serviço. Prioriza também tarefas e limitação de recursos aos usuários.
Acesso ao TOE	FTA	Gerencia sessões de usuário, estando relacionada a questões do tipo: • *A aplicação previne sessões concorrentes, terminadas ou bloqueadas após intervalos pré-definidos*? • *A aplicação inclui janela de login, definindo o uso apropriado do TOE*? • *A aplicação mostra históricos anteriores de login após sucesso da conexão*?
Canais e caminhos confiáveis	FTP	Lida com aspectos de canais de comunicação entre usuários e o produto, e entre o produto e outros produtos confiáveis de TI.

Uma família ligada aos requisitos funcionais de segurança compartilha objetivos de segurança, os quais podem-se diferenciar em ênfase ou rigor. Uma classe de requisitos define pesos de segurança para cada requisito, dependendo da criticidade do sistema e de sua necessidade de segurança. Por exemplo, a classe que trata da proteção dos dados do usuário (classe FDP – *User Data Protection*) possui entre suas famílias componentes com até quatro pesos, do FDP_ACF.1.1 ao FDP_ACF.1.4 (ACF – *Access Control Functions* - significa controle de acesso com base em atributos de segurança) (MOOKHEY, 2005).

De acordo com Albuquerque (2002), os requisitos funcionais de segurança definem as características funcionais da aplicação, no que diz respeito ao comportamento seguro do sistema.

Informações adicionais sobre os requisitos funcionais de segurança podem ser obtidas em ISO/IEC 15408-2 (2005b).

3.3.1.2 Requisitos de Garantia de Segurança

Os requisitos de garantia de segurança abordam os níveis crescentes de garantia e confiança que clientes depositam em produtos de TI.

Os elementos de garantia de segurança são divididos em três tipos, cada qual representado por uma letra ligada à classificação do elemento (15408-3, 2005c):

1) **D:** ação de garantia a ser executada pelo desenvolvedor do produto.
2) **C:** explica aquilo que deve ser demonstrado para garantir a evidência.
3) **E:** ação a ser tomada ou análise a ser executada pelo avaliador para confirmar que as evidências foram comprovadas e demonstradas.

A seguir, apresentam-se brevemente os requisitos de garantia de segurança e suas famílias:

- *Gerência de configuração e controle de versão* (Classe ACM): Automação da gerência; Escopo da gerência; Capacidade de gerência de configuração.

- *Entrega e operação do produto* (Classe ADO): Entrega; Procedimentos de instalação, geração e inicialização.
- *Desenvolvimento* (Classe ADV): Especificação funcional; Projeto de alto nível; Projeto de baixo nível; Representação da implementação; Correspondência entre as representações; Modelagem da política de segurança.
- *Documentação* (Classe AGD): Documentação de ajuda ao administrador; Documentação de ajuda ao usuário.
- *Suporte ao ciclo de vida* (Classe ALC): Segurança no desenvolvimento; Correção de falhas; Definição do ciclo de vida; Ferramentas e técnicas.
- *Testes* (Classe ATE): Cobertura; Profundidade; Testes funcionais de segurança; e Testes independentes.
- *Avaliação de vulnerabilidades* (Classe AVA): Análise de canais ocultos (*covert channel*); Uso inapropriado; Força das funções de segurança do aplicativo; Análise de vulnerabilidades.
- *Avaliação do Perfil de Proteção* (Classe APE): Descrição do produto; Ambiente de segurança; Introdução sobre o PP; Objetivos de segurança; Requisitos de segurança de TI; Requisitos de segurança de TI definidos explicitamente.
- *Avaliação do Alvo de Segurança* (Classe ASE): Descrição do produto; Ambiente de segurança; Introdução sobre o ST; Objetivos de segurança; Informações do PP; Requisitos de segurança de TI; Requisitos de segurança de TI definidos explicitamente; Resumo das especificações do produto.

Uma família ligada aos requisitos de garantia de segurança também compartilha objetivos de segurança os quais podem se diferenciar em ênfase ou rigor. Neste contexto, uma classe de requisitos de garantia igualmente define pesos de garantia de segurança para cada requisito, dependendo do nível de garantia escolhido para o produto. Por exemplo: a classe de gerência de configuração (Classe ACM) possui, entre suas famílias, componentes com até cinco pesos.

A filosofia da ISO/IEC 15408 afirma que uma maior garantia de segurança resulta na aplicação de maior esforço de avaliação, e o objetivo é aplicar o menor esforço requerido para fornecer o nível de garantia necessário. Um nível crescente de esforço é baseado em:

- *Escopo*: o tamanho do esforço é proporcional à porção do produto de software que é considerada.

- *Profundidade*: o tamanho do esforço é proporcional ao nível de detalhe do projeto e de implementação a ser inspecionado.
- *Rigor*: o tamanho do esforço é proporcional à maneira formal e estruturada que é aplicada.

Assim sendo, a exigência do ambiente de desenvolvimento é ditada pelo nível de garantia de segurança utilizado (ALBUQUERQUE, 2002). O nível EAL-3, por exemplo, é o nível mais alto de segurança que pode ser obtido sem um comprometimento significativo de custos e prazos (ISO/IEC 15408, 2005c).

3.3.2 Organização dos Níveis de Garantia de Segurança

Cada nível de garantia de segurança (EAL) possui um conjunto específico de componentes. O conjunto de níveis representa uma escala ascendente de rigor nos controles e nos requisitos de garantia, implicando um custo maior para cada nível até um determinado ponto, cuja viabilidade não seja mais aceitável.

A seguir, descreveremos os níveis EAL3 e EAL4, que são considerados aceitáveis em termos de garantia de segurança e viabilidade financeira.

3.3.2.1 EAL3 – Metodicamente Testado e Verificado

O EAL3 permite ao desenvolvedor obter o máximo de garantia através de uma engenharia de segurança na fase de projeto do sistema, sem uma grande modificação das práticas de desenvolvimento. Representa um nível moderado de garantia de segurança, necessitando uma análise completa da aplicação e de seu desenvolvimento, sem exigir reengenharia.

Segundo Albuquerque (2002), os requisitos de garantia exigidos para obter um produto de software compatível com este nível incluem:

- *Número de versão* (ACM_CAP.1): necessita ser facilmente identificado, ser único e não ambíguo, a fim de facilitar a identificação de problemas, caso ocorram.
- *Itens de configuração* (ACM_CAP.2): um gerenciamento de configuração precisa ser utilizado pelo desenvolvedor, para permitir a identificação dos itens controlados por ele.
- *Controle de mudanças* (ACM_CAP.3): um sistema de gerência de configuração precisa ser utilizado pelos desenvolvedores, para controlar mudanças em um código ou componente.

- *Cobertura dos artefatos da aplicação* (ACM_SCP.1): há um plano de gerência de configuração indicando os itens a serem submetidos à gerência de configuração.
- *Procedimentos de entrega* (ADO_DEL.1): definição de como o cliente recebe o sistema, incluindo novas versões, de forma a garantir sua origem e conformidade.
- *Procedimentos de instalação, geração, e inicialização* (ADO_IGS.1): definição de como instalar, gerar e iniciar o sistema de forma segura, incluindo requisitos de segurança do ambiente.
- *Demonstração informal de correspondência* (ADV_RCR.1): demonstração de que o *design* do sistema deve estar relacionado a atributos de segurança.
- *Projeto descritivo de alto nível* (ADV_HLD.1): descreve, de forma genérica, o funcionamento e como foram implementadas as funções de segurança do sistema.
- *Projeto com ênfase na segurança* (ADV_HLD.2): descreve o propósito, as exceções e os erros esperados de cada subsistema ou interface da aplicação.
- *Especificação funcional informal* (ADV_FSP.1): descreve o funcionamento esperado do sistema e de suas interfaces em relação à segurança.
- *Documentação de ajuda do usuário* (AGD_USR.1): Descreve os procedimentos para uso seguro do sistema por seus usuários.
- *Documentação de ajuda ao administrador* (AGD_ADM.1): apresenta os procedimentos para uso seguro do sistema por seus administradores.
- *Identificação de medidas de segurança* (ALC_DVS.1): mostra para todo o ciclo de vida da aplicação as medidas de ambiente e processos necessários para manter sua segurança.
- *Teste funcional* (ATE_FUN.1): realização pelo desenvolvedor de testes internos no sistema, fornecendo planos de teste e resultados obtidos.
- *Evidência de cobertura* (ATE_COV.1): demonstração pelo desenvolvedor de que os planos de teste e resultados obtidos cobrem todas as funcionalidades de segurança necessárias.
- *Análise da cobertura* (ATE_COV.2): análise pelo desenvolvedor da correspondência entre os itens, demonstrando que os planos de teste e resultados obtidos cobrem todas as funcionalidades de segurança necessárias.
- *Teste do projeto de alto nível* (ATE_DPT.1): análise pelo desenvolvedor de que os resultados obtidos são suficientes para atestar o correto funcionamento dos componentes de segurança.

- *Capacidade para teste independente* (ATE_IND.1): o sistema deve estar passível de ser testado em laboratório ou por entidade capacitada e reconhecida tanto pelo desenvolvedor quanto pelo cliente.
- *Teste independente por amostragem* (ATE_IND.2): submissão do sistema pelo desenvolvedor a testes independentes por amostragem, isto é, através da verificação de um subconjunto aleatório das funções de segurança.
- *Análise de vulnerabilidade por parte do desenvolvedor* (AVA_VLA.1): realização de uma análise de vulnerabilidade pelo desenvolvedor, que consiste em testar os métodos mais comuns de ataques contra a aplicação, documentando quais testes foram feitos, de que forma e com quais resultados.
- *Avaliação da força das funções de segurança* (AVA_SOF.1): geração pelo desenvolvedor de documento com cálculos para demonstrar que o tempo médio de quebra das chaves de criptografia, senhas e *hash* por força bruta é substancialmente superior ao tempo médio de troca dessas chaves e senhas.
- *Análise da documentação de ajuda* (AVA_MSU.1): fornecimento pelo desenvolvedor de documentação de ajuda que indique todos os modos de operação do sistema, inclusive aqueles de degradação ou de erro, e as medidas que devem ser tomadas para manter a segurança em cada um dos modos.

3.3.2.2 EAL4 – Metodicamente Projetado, Testado e Revisado

O EAL4 permite ao desenvolvedor obter o máximo de garantia através de uma engenharia de segurança baseada em boas práticas comerciais de desenvolvimento as quais, embora rígidas, não requerem conhecimento especializado, habilidades e outros recursos. É o nível mais alto considerado economicamente viável a ser aplicado em um sistema de informação.

O EAL4 é aplicável em circunstâncias onde desenvolvedores ou usuários necessitam de um nível moderado a alto de garantia de segurança em sistemas, estando preparado para incluir custos adicionais de engenharia de segurança.

Segundo Albuquerque (2002), os requisitos de garantia exigidos para obter um produto de software compatível com esse nível incluem:

- *Automação parcial da gerência de configuração* (ACM_AUT.1): o desenvolvedor precisa usar um sistema informatizado para cuidar da gerência de configuração.

- *Versionamento* (ACM_CAP.1): o número de versão do sistema precisa ser facilmente identificado, ser único e não ambíguo, a fim de facilitar a identificação de problemas, caso ocorram.
- *Itens de configuração* (ACM_CAP.2): um sistema de gerência de configuração precisa ser utilizado pelo desenvolvedor, para permitir a identificação dos itens controlados por ele.
- *Controle de mudanças* (ACM_CAP.3): um sistema de gerência de configuração também precisa ser utilizado pelos desenvolvedores, para controlar mudanças em um código ou componente.
- *Procedimentos de aceitação* (ACM_CAP.4): além do plano de gerência de configuração, deve haver procedimentos claros de geração da aplicação e de aceitação desta para liberação de versão.
- *Cobertura dos artefatos da aplicação* (ACM_SCP.1): deve haver cobertura, no mínimo, dos seguintes artefatos: plano de projeto, plano de testes, codificação da aplicação, manual do usuário, e plano de gerência de configuração.
- *Cobertura do controle de mudanças* (ACM_SCP.2): o gerenciamento de configuração deve ser capaz de associar o erro ou problema reportado às alterações no código realizadas para corrigi-lo.
- *Procedimentos de entrega* (ADO_DEL.1): definição de como o cliente recebe o sistema, incluindo novas versões, para garantir sua origem e conformidade.
- *Detecção de modificações na entrega* (ADO_DEL.2): o método usado para entrega do sistema ao cliente deve permitir a detecção de alterações efetuadas, que não tenham sido acordadas com o cliente.
- *Procedimentos de instalação, geração e iniciação* (ADO_IGS.1): definição clara de como instalar e iniciar o sistema de forma segura, incluindo requisitos de segurança do ambiente.
- *Demonstração informal de correspondência* (ADV_RCR.1): demonstração que permita associar modelagens de projeto aos atributos de segurança.
- *Modelo de política de segurança informal* (ADV_SPM.1): elaboração de políticas que descrevam as diretrizes de segurança seguidas pelo projeto e a implementação do sistema ao traduzir a especificação de segurança.

- *Projeto descritivo de alto nível* (ADV_HLD.1): documento que descreve, de forma genérica, o funcionamento e como foram implementadas as funções de segurança do sistema.
- *Projeto com ênfase na segurança* (ADV_HLD.2): deve descrever o propósito, as exceções e os erros esperados de cada subsistema ou interface da aplicação.
- *Projeto descritivo de baixo nível* (ADV_LLD.1): descreve o funcionamento e como foram implementadas as funções de segurança do sistema, apenas em baixo nível.
- *Implementação mínima de funcionalidades de segurança* (ADV_IMP.1): deve ser gerado um documento que descreva, para um subconjunto predefinido, a implementação realizada em um nível de detalhe, tal que nenhuma decisão precise ser tomada entre esse documento e a efetiva codificação das funções (nível de implementação).
- *Especificação funcional informal* (ADV_FSP.1): descreve o funcionamento esperado do sistema e de suas interfaces em relação à segurança.
- *Definição completa das interfaces externas* (ADV_FSP.2): descreve o funcionamento esperado do sistema e suas interfaces, do ponto de vista da segurança, cobrindo todas as interfaces externas do sistema.
- *Documentação de ajuda do usuário* (AGD_USR.1): descreve os procedimentos para uso seguro do sistema por seus usuários.
- *Documentação de ajuda ao administrador* (AGD_ADM.1): descreve os procedimentos para uso seguro do sistema por seus administradores.
- *Identificação de medidas de segurança* (ALC_DVS.1): descreve para todo o ciclo de vida da aplicação as medidas de ambiente e processos necessários a manter a sua segurança.
- *Modelo do ciclo de vida definido pelos desenvolvedores* (ALC_LDC.1): descreve o ciclo de vida do sistema, com manutenção, alterações, entrega, etc., além de pontos de controle do processo.
- *Ferramentas de desenvolvimento bem definidas* (ALC_TAT.1): descreve todas as ferramentas usadas para o desenvolvimento dos sistemas, além das opções de configuração e modos de operação dessas ferramentas.
- *Teste funcional* (ATE_FUN.1): o desenvolvedor deve realizar testes internos no sistema e fornecer os planos de teste e resultados obtidos.
- *Evidência de cobertura* (ATE_COV.1): o desenvolvedor deve demonstrar que os planos de teste e resultados obtidos cobrem todas as funcionalidades de segurança necessárias.

- *Análise da cobertura* (ATE_COV.2): o desenvolvedor deve analisar a correspondência entre os itens e demonstrar que os planos de teste e resultados obtidos cobrem todas as funcionalidades de segurança necessárias.
- *Teste do projeto de alto nível* (ATE_DPT.1): o desenvolvedor deve analisar se os resultados obtidos são suficientes para atestar o correto funcionamento dos componentes de segurança.
- *Capacidade para teste independente* (ATE_IND.1): o desenvolvedor deve permitir que o sistema seja testado, fornecendo os documentos acima especificados, bem como o próprio sistema, para o teste por laboratório ou entidade capacitada e reconhecida tanto pelo desenvolvedor quanto pelo cliente.
- *Teste independente por amostragem* (ATE_IND.2): o desenvolvedor deve submeter o sistema a testes independentes por amostragem, ou seja, através da verificação de um subconjunto aleatório das funções de segurança.
- *Análise de vulnerabilidade por parte do desenvolvedor* (AVA_VLA.1): o desenvolvedor deve realizar análise de vulnerabilidade, que consiste em testar os métodos mais comuns de ataques contra a aplicação, documentando quais testes foram feitos, de que forma e com quais resultados.
- *Análise de vulnerabilidade por terceiros* (AVA_VLA.2): o desenvolvedor deve encaminhar para avaliação em um laboratório ou órgão competente independente, reconhecido pelo desenvolvedor e pelo cliente, para uma análise de vulnerabilidade.
- *Avaliação da força das funções de segurança* (AVA_SOF.1): o desenvolvedor deve gerar um documento com os cálculos para demonstrar que o tempo médio de quebra das chaves de criptografia, senhas e *hash* por força bruta é substancialmente superior ao tempo médio de troca dessas chaves e senhas.
- *Análise da documentação de ajuda* (AVA_MSU.1): o desenvolvedor deve fornecer documentação de ajuda que indique todos os modos de operação do sistema, inclusive aqueles degradados ou de erro e as medidas que devem ser tomadas para manter a segurança em cada um dos modos.
- *Validação da análise* (AVA_MSU.2): o desenvolvedor deve fornecer uma análise que garanta a completude da avaliação de mau uso do sistema.

3.3.3 Estabelecer Segurança do Ambiente

Para estabelecer a segurança do ambiente, é preciso definir os ativos a serem manipulados pelo sistema (ativos que necessitam de proteção), e os objetivos do sistema.

Assim, a segurança do ambiente, onde o sistema deverá operar, é estabelecida (ISO/IEC 15408-1, 2005a).

3.3.4 Estabelecer Objetivos de Segurança

Segundo a ISO/IEC 15408-1 (2005a), estabelecer objetivos de segurança constitui-se de uma base acertada com o cliente do que a aplicação terá em termos de segurança. Envolve o levantamento de necessidades de segurança na forma de premissas, de ameaças, e de questões legais, normas internas e políticas de segurança. Um objetivo de segurança deve ser ligado a pelo menos uma ameaça ou política.

3.3.5 Estabelecer Requisitos de Segurança

De acordo com a ISO/IEC 15408-1 (2005a), estabelecer requisitos de segurança implica em verificar que requisitos ou controles de segurança atendem aos objetivos de segurança. É possível que um objetivo gere mais de um requisito.

Os requisitos de segurança devem incluir tanto os requisitos relacionados à existência de comportamentos desejados, quanto os requisitos relacionados à ausência de comportamentos indesejados. Uma fonte de requisitos de segurança é a parte 2 da ISO/IEC 15408 (2005b).

3.3.6 Estabelecer a Revisão das Especificações do Sistema

Consoante a ISO/IEC 15408-1 (2005a), os requisitos funcionais de segurança estabelecidos para o sistema devem ser revisados. É também definido o nível de garantia de segurança (seção 3.3.2) e seus requisitos de garantia, segundo a ISO/IEC 15408-3 (2005c). Os requisitos são analisados e verificados para garantir o nível definido.

3.3.7 Os Testes de Segurança (Classe ATE)

Apesar de o objetivo de um teste ser de fornecer evidências de que os requisitos foram satisfeitos, sabe-se que os testes por si só não garantem que a aplicação realize apenas as funções que ela se propõe. Assim, é possível que comportamentos não previstos sempre poderão existir.

Em geral, as famílias da classe de testes estão ligadas a outras famílias, logo que há necessidade de informações fornecidas pelos desenvolvedores da aplicação. De acordo com a abordagem em ISO/IEC 15408-3 (2005c), avaliadores são considerados testadores independentes do sistema.

3.3.7.1 Cobertura dos Testes (ATE_COV)

O objetivo desta família está relacionado com a completude da cobertura do teste. Trata-se de uma extensão com a qual os testes são realizados e se são ou não suficientes para demonstrar que o aplicativo opera como especificado.

Em especial, o componente Evidência da Cobertura (ATE_COV.1) tem como objetivo estabelecer que as funcionalidades de segurança foram testadas em relação à sua especificação funcional. Isto é conseguido pelo exame entre o que foi testado e a especificação funcional.

3.3.7.2 Profundidade dos Testes (ATE_DPT)

Os componentes desta família tratam do nível de detalhe com o qual as funcionalidades de segurança são testadas. O teste das funcionalidades de segurança é baseado no aumento da profundidade das informações derivadas das análises.

O objetivo é restringir os riscos de não detectar um erro durante o desenvolvimento do produto de software. Além disso, os componentes dessa família, uma vez que o teste é baseado em estruturas internas do produto, estão mais propensos a descobrir qualquer código malicioso que foi inserido.

A profundidade dos testes garante não apenas que as funcionalidades de segurança exibam o comportamento externo de segurança desejado, mas também que este comportamento se origine de mecanismos internos de segurança que operam corretamente.

3.3.7.3 Testes Funcionais (ATE_FUN)

Testes funcionais realizados pelo desenvolvedor estabelecem que as funcionalidades de segurança exibem as propriedades necessárias para satisfazer os requisitos funcionais do sistema. Garantem que as funcionalidades de segurança satisfazem, no mínimo, os requisitos funcionais de segurança, embora não se possa afirmar que as funcionalidades de segurança não façam mais do que foi especificado.

Os testes funcionais não estão limitados a resultados positivos que atestam a presença das funções de segurança pedidas. Porém, podem também incluir testes negativos para checar pela ausência de comportamentos indesejados, normalmente baseados na inversão dos requisitos funcionais.

3.3.7.4 Testes Independentes (ATE_IND)

Essa família especifica que os testes podem ser realizados no sistema por terceiros com conhecimento especializado e que não participaram do desenvolvimento.

Seu objetivo é demonstrar que as funções de segurança operam como especificado. Todavia, há uma necessidade de restringir o risco de uma avaliação incorreta por parte da equipe de desenvolvimento, que resulte na incorreta implementação das especificações, ou ignora código não aderente com as especificações.

3.3.8 Avaliação de Vulnerabilidades (Classe AVA)

A última classe de garantia de segurança da ISO/IEC 15408 relaciona-se com processo de avaliação e análise de vulnerabilidades, o possuem as famílias a seguir.

3.3.8.1 Análise de Canal de Cobertura (AVA_CCA)

Canal de cobertura (*Covert channel*) pode ser definido como um canal oculto com fluxo ilícito de informação. A análise desses canais almeja descobrir sua existência e a capacidade de esses canais serem explorados.

A validação da análise de canais permite ao avaliador verificar aspectos como a identificação, a estimativa de capacidade, o monitoramento, e a análise de cenários de exploração dos canais.

3.3.8.2 Uso Inapropriado (AVA_MSU)

O objetivo dessa família é investigar se o sistema pode ser configurado ou utilizado de maneira insegura, mesmo os administradores e usuários acreditando que tudo está seguro. Seus objetivos são:

- Minimizar a probabilidade de configurar ou instalar o sistema de maneira que seja inseguro, sem que o usuário e o administrador sejam capazes de detectar o problema.
- Minimizar o risco de erros operacionais humanos ou que não causem desativação, desabilitação ou falha em ativar as funções de segurança, resultando em um estado inseguro não detectável.

Em especial, o componente *Exame da Documentação de Apoio* (AVA_MSU.1) tem como objetivo garantir que não haja orientação errônea, irracional ou conflituosa na documentação de apoio, e que procedimentos de segurança para todos os modos de operação tenham sido identificados. Estados inseguros, onde o sistema possa estar, deveriam ser fáceis de detectar.

3.3.8.3 Força das Funções de Segurança do Aplicativo (AVA_SOF)

Mesmo que as funções de segurança do sistema não possam ser contornadas, desativadas ou corrompidas, pode ainda ser possível afetá-las, porque há ainda vulnerabilidades escondidas nos mecanismos de segurança. Para estas funções, a qualificação de seus comportamentos seguros pode ser obtida utilizando-se resultados quantitativos ou estatísticos do comportamento de segurança desses mecanismos e o esforço necessário para contorná-los.

O único componente da família, *Avaliar a Força das Funções de Segurança* (AVA_SOF.1), faz com que os desenvolvedores executem um teste de força nas funcionalidades de segurança, para cada mecanismo de segurança que possua alguma afirmação de força em sua especificação (ALBUQUERQUE, 2002). Por exemplo: criptografia RSA de 128 bits.

Os testes devem indicar se o mecanismo de segurança atinge ou ultrapassa sua força. O avaliador verifica as evidências apresentadas e sua coerência, isto é, verifica se as afirmações de força exigidas estão corretas.

3.3.8.4 Análise de Vulnerabilidades (AVA_VLA)

Análise de vulnerabilidade é uma avaliação que determina se vulnerabilidades identificadas durante o ciclo de vida poderiam permitir que usuários violassem as funcionalidades de segurança.

A análise de vulnerabilidade lida com as ameaças de que o usuário descubra falhas no sistema, que permitam acesso não autorizado aos recursos, interferir ou alterar as funcionalidades de segurança, ou interferir com as autorizações de outros usuários.

O componente *Análise de Vulnerabilidade pelo Desenvolvedor* (AVA_VLA.1) tem como objetivo a realização de análise de vulnerabilidade por parte do desenvolvedor, de forma a identificar vulnerabilidades de segurança óbvias, e confirmar que elas não possam ser exploradas no ambiente onde o sistema será instalado.

O componente *Análise de Vulnerabilidade por Terceiros* (AVA_VLA.2) tem como objetivo a realização de análise de vulnerabilidade por parte de entidade independente, de forma a identificar vulnerabilidades de segurança óbvias, e confirmar que elas não podem ser exploradas no ambiente onde o sistema será instalado.

Um avaliador independente executa teste de invasão, baseado na documentação do desenvolvedor, para determinar se o produto resiste a ataques realizados por invasores com

pouco conhecimento. Dessa forma, o avaliador valida a análise feita pelo desenvolvedor, sendo possível até descobrir novas vulnerabilidades não percebidas anteriormente.

A questão de segurança envolve saber quanto risco uma empresa pretende assumir a ponto de solucionar eficazmente o problema. (McGRAW, 2000a). O OCTAVE é uma técnica estratégica de planejamento e avaliação baseada em riscos para segurança.

3.4 OCTAVE

OCTAVE (*The Operationally Critical Threat, Asset, and Vulnerability Evaluation*) é um processo auto orientado, isto é, pessoas da organização assumem a responsabilidade de configurar a estratégia de segurança da própria organização.

Este processo influencia as pessoas a utilizarem seu conhecimento das práticas e processos de segurança da organização, para capturar seu estado atual de segurança. Os riscos dos ativos mais críticos são usados para priorizar áreas que necessitam de melhoria e para definir a estratégia para a organização.

OCTAVE é direcionado para o risco organizacional, focando em estratégia e problemas práticos. Quando se aplica este processo, pequenas equipes das unidades operacionais (ou de negócio) e da área de tecnologia da informação (TI) podem trabalhar juntas, para tratar as necessidades de segurança da organização, balanceando três aspectos chave: risco operacional, práticas de segurança, e tecnologia.

O processo OCTAVE é guiado por dois desses aspectos: risco operacional e práticas de segurança. A tecnologia é examinada apenas em relação às práticas de segurança, permitindo a uma organização refinar a visão de suas práticas atuais de segurança.

Ao usar o processo OCTAVE, uma organização toma decisões de proteção da informação baseadas em riscos de confidencialidade, integridade, e disponibilidade dos ativos críticos de informação. Todos os aspectos do risco (ativos, ameaças, vulnerabilidades, e impacto organizacional) são fatorados em tomadas de decisão, permitindo que a organização ajuste a estratégia de proteção de seus riscos de segurança.

3.4.1 Características Chave do Processo OCTAVE

OCTAVE é auto orientado. Isto requer que uma organização gerencie o processo de avaliação e tome decisões para proteger as informações. Uma equipe interdisciplinar, chamada de equipe de análise, conduz a avaliação. A equipe inclui pessoas das unidades de negócio e área de TI, pois ambas as perspectivas são importantes, quando se caracterizam a visão global e organizacional do risco de segurança da informação.

OCTAVE é um processo de avaliação baseado nos ativos. Sua equipe de análise:

- identifica ativos de informação, que sejam importantes à organização;
- foca as atividades de análise de risco nos ativos julgados como os mais críticos à organização;
- considera os relacionamentos entre os ativos críticos, as ameaças a estes ativos, e as vulnerabilidades (tanto organizacional quanto tecnológica), que podem expor os ativos às ameaças;
- avalia riscos em um contexto operacional – como os ativos são manipulados para conduzir o negócio da organização e como estes ativos ficam em risco devido às ameaças de segurança.
- cria uma estratégia de proteção prática para a melhoria organizacional, bem como planos de mitigação de risco, para reduzir o risco contra os ativos críticos de organização.

Os aspectos organizacional, tecnológico, e de análise de uma avaliação de risco de segurança da informação são complementados em três fases. OCTAVE é organizado ao redor desses três aspectos básicos (Figura 3.1), permitindo à organização configurar de forma compreensível as necessidades de segurança da informação da organização. As fases são:

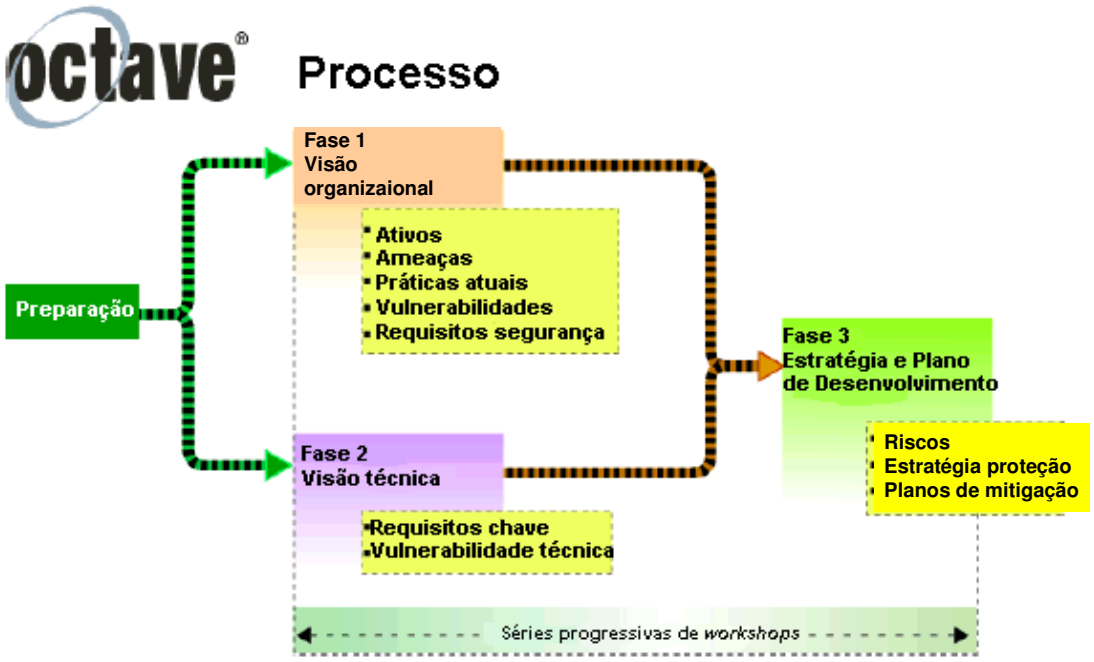

Figura 3.1: Fases do OCTAVE

Fase 1: Construir Perfis de Ameaça dos Ativos: é uma avaliação organizacional. A equipe de análise determina o que é importante para a organização (ativos de informação) e o que está atualmente sendo feito para proteger estes ativos. A equipe então seleciona aqueles ativos, que são mais importantes à organização, e descreve requisitos de segurança para cada um desses ativos críticos. Finalmente, são identificadas as ameaças de cada ativo crítico, criando um perfil de ameaça para aquele ativo.

Fase 2: Identificar Vulnerabilidades de Infra-estrutura: é avaliação da infra-estrutura da informação. A equipe de análise examina caminhos de acesso à rede, identificando classes de componentes de tecnologia de informação relacionados a cada ativo crítico. A equipe determina a extensão a que cada classe de componente resiste a ataques à rede.

Fase 3: Desenvolver Planos e Estratégias de Segurança: durante esta parte da avaliação, a equipe de análise identifica riscos contra os ativos críticos da organização e decide o que fazer em relação a eles. A equipe cria uma estratégia de proteção para a organização e planos de mitigação para tratar os riscos contra os ativos críticos, baseados em análises recolhidas da informação.

3.4.2 Critérios do OCTAVE

Os elementos essenciais, ou requisitos, do processo OCTAVE, são agrupados em um conjunto de critérios. Pode haver muitos métodos consistentes com esses critérios, mas há apenas um conjunto de critérios OCTAVE. Neste ponto, dois métodos consistentes com o critério foram desenvolvidos. O primeiro, Método OCTAVE, foi projetado para ser aplicado em grandes organizações. O Segundo, chamado OCTAVE-S (ALBERTS, 2003), foi desenvolvido para pequenas organizações. Em adição, outros podem definir métodos para contextos específicos que sejam consistentes com o critério.

O critério do OCTAVE é formado por um conjunto de princípios, atributos, e saídas. Princípios são os conceitos fundamentais que guiam a natureza da avaliação, e definem a filosofia por trás do processo de avaliação. Eles modelam a abordagem de avaliação e fornecem a base para o processo de avaliação. Por exemplo, *auto direção* é um dos princípios do OCTAVE. O conceito da auto direção significa que as pessoas de dentro da organização estão na melhor posição para liderar a avaliação e tomar as decisões.

Os requisitos da avaliação são agrupados nos atributos e saídas. Atributos são as características da avaliação. Eles são os requisitos que definem os elementos básicos do processo OCTAVE e definem o que é necessário para tornar a avaliação um sucesso em ambas as perspectivas de processo e organizacional. Atributos são derivados dos princípios do

OCTAVE. Por exemplo: um dos atributos do OCTAVE é que uma equipe interdisciplinar (a equipe de análise), formada por pessoal da própria organização, conduza a avaliação. O princípio por trás da criação da equipe de análise é a auto direção.

Finalmente, as saídas são os resultados requeridos de cada fase da avaliação. Eles definem os objetivos que a equipe de análise deve alcançar durante cada fase. Há mais de um conjunto de atividades que podem produzir as saídas do OCTAVE. Por esta razão, um único conjunto de atividades não é especificado. As saídas são organizadas de acordo com as três fases acima mencionadas.

3.4.3 OCTAVE Como Parte de um Processo Contínuo

Durante o processo OCTAVE, a equipe de análise realiza atividades para:

- identificar os riscos de segurança da informação da organização;
- analisar os riscos para determinar as prioridades;
- planejar para melhorar através do desenvolvimento de uma estratégia de melhoria organizacional e planos de mitigação de risco, para reduzir os riscos contra os ativos críticos da organização.

Após implementar o OCTAVE, a equipe de análise, ou outro pessoal designado:

- planeja como implementar a estratégia de proteção e planos de mitigação do risco através do desenvolvimento detalhado de planos de ação (isto pode incluir uma análise detalhada de custo-benefício entre as estratégias e ações, e resultar em planos detalhados de implementação);
- implementa planos detalhados de ação;
- monitora planos de ação por prazo e por eficácia (inclui monitoramento de riscos em quaisquer mudanças);
- controla variações no plano de execução tomando ações corretivas apropriadas.

Uma avaliação de risco de segurança da informação faz parte das atividades de uma organização para gerenciar os riscos de segurança da informação. OCTAVE é uma atividade de avaliação, não um processo contínuo. Assim, ele tem um início e fim definidos. A Figura 3.2 mostra o relacionamento entre estas atividades e onde o OCTAVE se posiciona, em que as atividades de gerência de risco definem o ciclo PDCA de Deming (2000) – Planejar, Fazer, Controlar, Agir (*Plan-Do-Check-Act*).

O tempo entre as avaliações pode ser pré-determinado ou ativado por eventos maiores (p.ex., reorganização da empresa ou redesenho da infra-estrutura de computação da organização). Entre as avaliações, uma organização pode periodicamente identificar novos

riscos, analisar esses riscos em relação aos riscos existentes, e desenvolver planos de mitigação para eles.

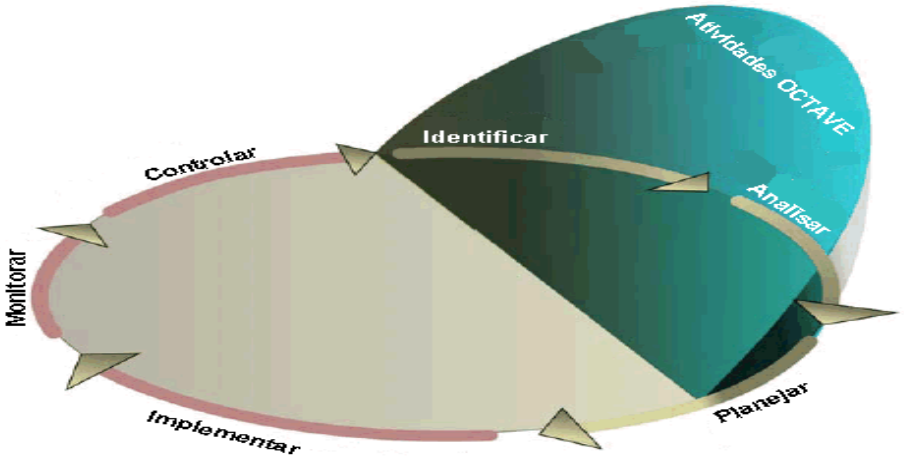

Figura 3.2: OCTAVE e atividades de gerência de risco

3.4.4 O Método OCTAVE

O Método OCTAVE compreende três fases requeridas pelo critério do OCTAVE. Os processos nestas fases são descritos abaixo:

Fase 1: Construir Perfis de Ameaça dos Ativos: agrupa informação de toda organização, definindo perfis de ameaça para os ativos críticos. Possui os seguintes processos:

- Processo 1: Identificar Conhecimento da Alta Gerência: coleta de informação sobre os ativos importantes, requisitos de segurança, ameaças, e processos atuais da organização que funcionam bem, e vulnerabilidades. A coleta é feita junto a um conjunto representativo de altos executivos.
- Processo 2: Identificar Conhecimento da Área Operacional: coleta de informação sobre ativos importantes, requisitos de segurança, ameaças, e processos atuais da organização que funcionam bem, e vulnerabilidades. A coleta é feita junto aos gerentes de áreas operacionais selecionadas.
- Processo 3: Identificar Conhecimento dos Técnicos: coleta de informação sobre ativos importantes, requisitos de segurança, ameaças, e processos atuais da organização que funcionam bem, e vulnerabilidades. A coleta é feita junto aos colaboradores técnicos e membros da TI de áreas operacionais selecionadas.
- Processo 4: Criar Perfis de Ameaça: seleciona de três a cinco ativos críticos de informação e define os perfis de ameaça para estes ativos.

Fase 2: Identificar Vulnerabilidades de Infra-estrutura: analisa e avalia os componentes chave dos sistemas, que apóiam os ativos críticos em busca de vulnerabilidades tecnológicas.

- Processo 5: Identificar Componentes Chave – Um conjunto representativo de componentes chave dos sistemas que suportam ou processam os ativos críticos de informação é identificado, e uma abordagem para avaliá-los é definida.
- Processo 6: Avaliar Componentes Selecionados – Ferramentas são executadas para avaliar os componentes selecionados, e os resultados são analisados para refinar os perfis de ameaça (para ameaças de acesso à rede) dos ativos críticos.

Fase 3: Desenvolver Planos e Estratégia de Segurança – O objetivo primário desta fase é avaliar os riscos dos ativos críticos e desenvolver uma estratégia de proteção organizacional e planos de mitigação dos riscos.

- Processo 7: Conduzir Análise de Riscos – Critérios de avaliação de impacto são definidos para estabelecer uma base comum para determinar o valor do impacto (alto, médio, ou baixo) devido às ameaças contra os ativos críticos. Todos os riscos têm seu impacto avaliado.
- Processo 8: Desenvolver Estratégia de Proteção – A equipe desenvolve uma ampla estratégia de proteção organizacional focada em melhorar as práticas de segurança da organização bem como planos de mitigação para reduzir os principais riscos contra os ativos críticos.

3.4.5 OCTAVE-S

OCTAVE-S é uma variação do OCTAVE adaptada para os recursos limitados e restrições peculiares tipicamente encontrados em pequenas organizações (menos que 100 pessoas). Ela foi projetada para organizações que podem disponibilizar uma equipe de três a cinco pessoas para conduzir todas as atividades de avaliação, sem a necessidade das atividades formais de coleta de dados (ALBERTS, 2003).

Outra diferenciação do OCTAVE-S relaciona-se com a Fase 2 de avaliação da infra-estrutura de computação. Pequenas organizações freqüentemente terceirizam, em parte ou em sua totalidade, a manutenção dos seus sistemas de computador. Então, a Fase 2 do OCTAVE-S é uma inspeção e revisão abreviada dos processos usados para dar segurança à infra-estrutura de computação da organização.

OCTAVE-S tem as mesmas três fases descritas na abordagem OCTAVE e no Método OCTAVE. Contudo, os processos são de alguma forma diferentes do Método OCTAVE.

Fase 1: Construir Perfis de Ameaça dos Ativos – Durante esta fase, informação organizacional é identificada e utilizada para definir perfis de ameaça de três a cinco ativos críticos de informação.

- Processo S1: Identificar Informação Organizacional – A equipe de análise identifica os ativos de informação que são importantes para a organização, define um conjunto de critérios de avaliação de impacto, e define o estado atual das práticas de segurança da organização.
- Processo S2: Criar Perfis de Ameaça – A equipe de análise seleciona de três a cinco ativos críticos de informação e define os requisitos de segurança e perfis de ameaça contra estes ativos.

Fase 2: Identificar Vulnerabilidades de Infra-estrutura – Durante esta fase, a equipe de análise realiza uma revisão superficial de sua infra-estrutura e práticas tecnológicas para refinar os perfis de ameaça.

- Processo S3: Examinar a Infra-estrutura de Computação em Relação aos Ativos Críticos – A equipe de análise analisa os caminhos de acesso nos sistemas que apóiam os ativos críticos e determina quão bem seus processos tecnológicos estão protegendo estes ativos.

Em vez de usar dados de vulnerabilidade para refinar sua visão das práticas atuais de segurança, uma organização conduzindo uma avaliação OCTAVE-S examina os processos empregados para configurar e manter de forma segura sua infra-estrutura de computação. Quaisquer deficiências na capacidade organizacional são anotadas e consideradas durante a Fase 3, quando a organização desenvolve sua estratégia de proteção.

Fase 3: Desenvolver Planos e Estratégias de Segurança – Durante esta fase, os riscos contra os ativos críticos são avaliados e uma estratégia de proteção organizacional e planos de mitigação de risco são definidos.

- Processo S4: Identificar e Analisar Riscos – A equipe de análise avalia o impacto de todos os riscos e, opcionalmente, a probabilidade.
- Processo S5: Desenvolver Planos de Mitigação e Estratégias de Proteção – A equipe desenvolve planos de mitigação de risco e estratégia de proteção organizacionais baseados nas práticas de segurança.

3.4.6 Escolhendo Entre os Métodos

O Método OCTAVE está estruturado para uma equipe de análise com algum conhecimento de problemas de TI e de segurança, empregando uma abordagem aberta para

agrupar e analisar informação. Por outro lado, OCTAVE-S é mais estruturado. Conceitos de segurança são embutidos nas planilhas do OCTAVE-S, permitindo seu uso por praticantes com menor experiência. Equipes experientes podem achar o OCTAVE-S muito limitado, enquanto equipes inexperientes podem se perder ao usar o Método OCTAVE.

Devido ao fato de apenas dois métodos terem sido desenvolvidos pelo *Software Engineering Institute* (SEI), alguns usuários podem achar que nenhum método atende às suas necessidades. Muitos métodos que integram pedaços dos dois métodos para um tipo intermediário também são possíveis. Enquanto os métodos adaptados respeitarem o critério OCTAVE, eles ainda serão considerados avaliações OCTAVE.

Existe a possibilidade de se considerar o uso do OCTAVE-S em projetos individuais, nichos de negócio, ou departamentos, e agrupar a informação para obter uma perspectiva ampla da organização em vez de usar o Método OCTAVE. Apesar de esta abordagem ser teoricamente possível, não há experiência até o momento definindo como isto seria realizado.

Em relação à pesquisa, utilizou-se uma combinação do Método OCTAVE e do OCTAVE-S. Isso se justifica pela característica do OCTAVE-S de tratar com maior simplicidade a identificação das vulnerabilidades de infra-estrutura física, conforme evidenciado na fase 2 “Identificar Vulnerabilidades de Infra-estrutura” do OCTAVE-S, seção 3.4.5. Uma vez que a segurança física não é considerada minuciosamente na pesquisa, escolheu-se essa fase 2, em detrimento da fase homônima do Método OCTAVE para apoiar a identificação das atividades do Processo de Apoio à Segurança de Software.

Maiores informações sobre o Método OCTAVE e sobre o OCTAVE-S podem ser obtidas, respectivamente, em (ALBERTS, 2001) e (ALBERTS, 2003).

3.5 Conclusão

Este capítulo apresentou as abordagens de segurança da informação utilizadas para a concepção inicial de um processo contendo atividades de segurança para produzir software mais seguro.

O SSE-CMM é um modelo voltado para desenvolver as capacidades de engenharia de segurança. Sua adaptação para um padrão da ISO (2006) ratifica a importância do modelo. Como um modelo de maturidade, semelhante ao CMMI (2006), o SSE-CMM não explica como cada prática base pode ser implantada. Outra característica do modelo é que ele é genérico para atender a vários tipos de empresas, sendo , às vezes, complicado de entender. Além disso, para a finalidade do trabalho, o escopo do modelo era demasiado grande. Logo, apenas a parte cujo foco é a segurança foi utilizada.

A ISO/IEC 27002, anteriormente denominada ISO/IEC 17799, contempla um extenso conjunto de opções de controles de segurança a serem aplicados em diversos processos de empresas de vários nichos de atuação. Isso pode ser considerado uma fraqueza, pois essa característica generalista pode levar a interpretações equivocadas. A norma também não esclarece a melhor forma de implantar cada tipo de controle.

A ISO/IEC 15408 tem seu uso restrito, normalmente, entre grandes organizações que precisam ter garantias da segurança do software adquirido ou produzido. Isso é conseqüência de sua complexidade que dificulta a realização de avaliações por exigir conhecimento muito especializado. Isso o torna caro de aplicar e consome tempo maior. Outro problema é manter o foco individualmente em um produto de software (TOE), não considerando a interdependência com outros sistemas e componentes. Consoante Prieto-Díaz (2004), uma deficiência adicional é que a norma não se preocupa em verificar se os requisitos considerados no documento de requisitos são, de fato, atendidos pelo produto desenvolvido.

O processo OCTAVE foi escolhido por sua simplicidade em tratamento da análise de risco. Além disso, sua orientação voltada para a análise de risco de processo realizada pelas próprias pessoas responsáveis pelo processo ou que convivem com ele fazem do OCTAVE uma solução de alto custo-benefício e que produz resultados mais rápido.

A partir das especificidades de cada abordagem, foi realizado um estudo comparativo com o fito de identificar semelhanças entre as abordagens. O resultado do mapeamento serviu para montar a uma versão generalista do processo de apoio, então chamado de processo seguro.

Esse resultado do estudo foi organizado em uma tabela de mapeamento, Apêndice 1. Por questões de simplicidade, só estão representadas no mapeamento as práticas base do SSE-CMM que possuem correlação com, no mínimo, duas outras abordagens.

O próximo capítulo apresentará algumas abordagens de processos de software seguro.

Capítulo **4**

Trabalhos Relacionados a Processos Seguros

Este capítulo apresenta alguns trabalhos que tratam de práticas de segurança de software.

Será apresentado um conjunto de trabalhos relacionados com o foco da proposta desta pesquisa, que é promover um Processo de Apoio à Segurança de Software (Capítulo 6).

4.1 Processos para Produzir Software Seguro

O trabalho desenvolvido pelo *National Cyber Security Summit* e intitulado *Processes to Produce Secure Software* (DAVIS, 2004) apresenta um relatório que dissemina o uso de práticas conhecidas de desenvolvimento de software seguro.

4.1.1 Requisitos para Processo de Software Seguro

Para efetivamente produzir produtos de software seguros, um processo de desenvolvimento deve conter os seguintes requisitos:

- *Cobertura*: o processo deve cobrir todo o ciclo de vida de software desde a análise de requisitos através do projeto, codificação, entrega, manutenção, melhoria, e descontinuação.
- *Definição*: o processo deve ser precisamente definido de tal forma que possa ser ensinado, suportado, verificado, mantido, melhorado, e certificado. Além disso, todos os produtos e todas as atividades do processo devem ser precisos e compreensivelmente medidos.
- *Integridade*: o processo deve estabelecer e guardar a integridade do produto através do ciclo de vida, começando com os requisitos e projeto e incluindo gerência segura de configuração e entrega rigorosa.
- *Medidas*: o processo deve incluir medidas para verificar que os desenvolvedores do produto são coerentes e capazes de usar corretamente o processo. Deve ser verificado se o processo foi apropriadamente usado para desenvolver o produto, e que o processo produz consistentemente produtos de software seguros. Em adição, as medidas são requeridas para identificar propriedades e características de segurança relevantes do produto. As

medidas devem mostrar quão bem o processo foi utilizado e onde e quando ações corretivas são necessárias.

- *Rastreamento*: o processo deve permitir o rastreamento de tal forma que não comprometa a segurança dos produtos resultantes.
- *Usabilidade*: o processo deve ser utilizável por desenvolvedores qualificados, especialistas em segurança, gerentes, entre outros apropriadamente treinados, devendo ainda ser econômico e prático para desenvolver um amplo leque de tipos de sistemas.
- *Propriedade*: o processo deve ser proprietário, suportado, amplamente disponível e regularmente atualizado, para refletir mudanças nas condições de ameaças e melhorias no conhecimento e tecnologia.
- *Suporte*: o processo deve ser suportado por programas de conscientização e material de treinamento, guias de instrutor, ferramentas de apoio, e testes de qualificação.
- *Estado da arte*: o processo deve utilizar os melhores métodos para projeto, desenvolvimento, teste, medição de produto, e documentação do produto.

4.1.2 Aplicação do Processo Seguro de Software

O proprietário do processo habilita organizações a qualificar seus próprios especialistas no processo, a treinar sua própria equipe de sistema, e a gerenciar seu próprio processo de desenvolvimento. A organização deve ter a habilidade de monitorar e revisar seu próprio trabalho e verificar que seus produtos são seguros. Enquanto o treinamento e a qualificação dos desenvolvedores podem ser delegados a empresas de treinamento ou de consultoria que utilizam o processo, todo o trabalho de desenvolvimento e qualificação do produto deve ser rigorosamente controlado e rastreado.

O treinamento e a qualificação dos instrutores do processo e/ou revisores devem permanecer controlados e monitorados para garantir um alto grau de aderência. É ideal treinar todos os envolvidos em todas as etapas do desenvolvimento.

Deve-se também identificar e qualificar novos processos ou práticas que produzam sistemas seguros e que satisfaçam os requisitos definidos para que um processo produza software seguro.

4.1.3 Práticas Técnicas

Algumas vulnerabilidades de segurança são causadas por distrações que podem gerar alguns tipos de defeitos: erros de declaração, erros de lógica, erros de controle de laço, erros de expressões condicionais, falhas para validar entrada de dados, erros de especificação de interface, e erros de configuração. Essas causas estão relacionadas, em sua maioria, pelo uso de práticas de engenharia de software. No entanto, outras vulnerabilidades de segurança são causadas por problemas de modelagem, de arquitetura, e de projetos específicos de segurança, tais como: falha na identificação das ameaças, autenticação inadequada, autorização inválida, uso incorreto de criptografia, e falha na proteção de dados.

Práticas efetivas especializadas em segurança são necessárias para tratar estes problemas acima descritos. Práticas técnicas devem ser usadas em um contexto geral de processo planejado e gerenciado para produzir software seguro. O processo precisa planejar o uso de práticas, monitorar suas execuções e medir a eficácia.

A maioria das práticas descritas requer conhecimento de segurança. O auxílio de pessoas experientes em segurança é recomendável durante todas as fases do ciclo de vida do aplicativo, especialmente durante a especificação e o projeto.

Enquanto muitas práticas estão em uso para produzir software seguro, pouca evidência existe sobre a eficácia dessas práticas. Seguem algumas práticas técnicas sugeridas: modelagem de ameaças, árvores de ataque, guias e *checklist*, práticas de ciclo de vida, e gerência de riscos.

Modelagem de ameaças

Modelagem de ameaças é uma metodologia de análise de segurança que pode ser usada para identificar riscos, e guiar decisões de projeto, codificação, e teste. A metodologia é mais utilizada nas fases iniciais do projeto, envolvendo as ameaças com maior potencial de causar o máximo dano à aplicação.

Segundo Howard (2002), a modelagem de ameaças auxilia os desenvolvedores a entender onde o produto é mais vulnerável e, por conseguinte, a escolher as técnicas apropriadas para atenuá-las, produzindo sistemas mais seguros e com maior qualidade.

Em geral, modelagem de ameaças envolve a identificação dos principais ativos da aplicação, a decomposição da aplicação, identificação e categorização das ameaças para cada ativo ou componente, classificação das ameaças baseado em uma classificação do risco, e, por fim, o desenvolvimento de estratégias de redução de ameaças as quais são implementadas no projeto, código, e casos de teste.

Árvores de ataque

As árvores de ataque modelam o processo de decisão para realizar ataques. Ataques contra um sistema são representados em uma estrutura de árvore. A raiz representa a meta do atacante, como, por exemplo, roubar o número do cartão de crédito. Os nós representam ações que o atacante realiza e cada caminho na árvore representa um único ataque para atingir a meta do atacante.

Árvores de ataque podem ser usadas para responder questões do tipo: *Qual o ataque mais fácil*? *Qual o ataque mais barato*? *Qual o ataque que causa o maior dano*? *Qual o ataque mais difícil de detectar*? Árvores de ataque são utilizadas para apoiar a análise de riscos, para responder questões acerca da segurança do sistema, para capturar conhecimento de segurança de forma a reutilizá-lo, e para projetar, implementar e testar proteções contra os ataques.

Guias e *Checklists*

Os guias não devem ser confundidos com processos ou métodos. Não fornecem um passo-a-passo de como fazer algo, mas são princípios úteis a serem lembrados, quando se projetam sistemas.

Os *checklists* ajudam os desenvolvedores com listas de itens a serem checados ou lembrados. Seu uso deve ser garantido, sua eficácia medida, e devem ser atualizados baseados em sua eficácia.

Práticas de ciclo de vida

Davis (2004) apresenta uma visão geral de práticas de ciclo de vida para desenvolvimento de sistemas (Figura 4.1).

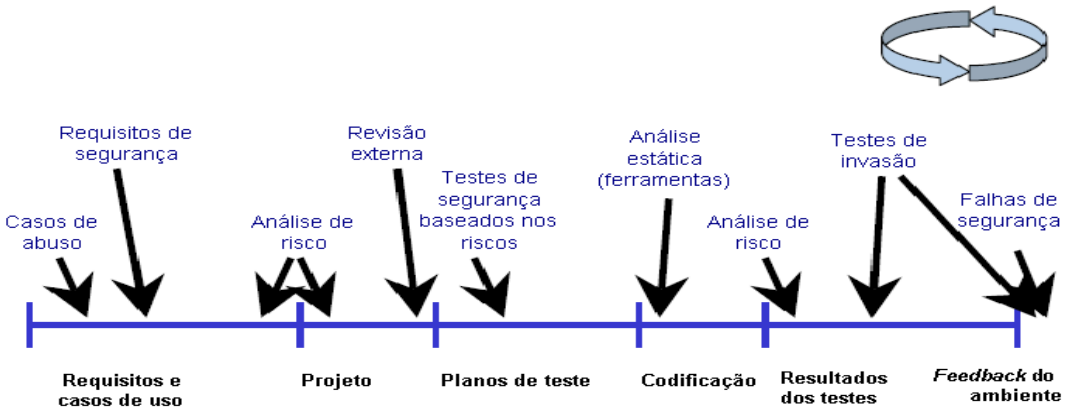

Figura 4.1: Melhores práticas de software seguro aplicadas a vários artefatos de software

Os requisitos de segurança envolvem a segurança funcional e características e propriedades de sistemas. Casos de abuso descrevem o comportamento do sistema sob ataque. Construí-los exige cobertura explicita do que deve ser protegido, de quem, e por quanto tempo (HOPE, 2004).

Tanto no estágio de especificação da arquitetura quanto no estágio de projeto hierárquico, a análise de riscos é uma necessidade onde os analistas de segurança podem descobrir e classificar os riscos para que possam ser mitigados (VERDON, 2004).

Teste de segurança envolve, principalmente, duas estratégias: teste da funcionalidade de segurança com técnicas padrões de teste funcional, e teste de risco de segurança baseado em padrões de ataque e modelos de ataque.

No estágio de codificação, é aconselhável o uso de ferramentas de análise estática. Isto é, ferramentas que verificam o código fonte a procura de vulnerabilidades comuns.

Segundo Chess (2004), ferramentas de análise estática examinam o texto do programa sem executá-lo, procurando por um conjunto fixo de padrões ou regras. Assim, se uma regra não for escrita para encontrar um problema particular, a ferramenta nunca encontrará o problema.

Os testes de invasão (*penetration tests*) têm como vantagem o bom entendimento do sistema no seu ambiente real. Contudo, qualquer teste de invasão de caixa preta, que não considere a arquitetura do sistema, falhará em descobrir algum risco. Um teste de invasão pode apenas identificar uma pequena amostra de todos os possíveis riscos de segurança no sistema. Para otimizar esta realidade, ferramentas devem fazer parte do teste de invasão (ARKIN, 2005).

A equipe de operação deve monitorar o comportamento dos sistemas para identificar brechas de segurança. O conhecimento adquirido com o entendimento dos ataques e brechas deve ser compartilhado com a equipe de desenvolvimento e implantado, de forma cíclica, no ciclo de vida.

Gerenciamento de riscos

Cada vez mais, produzir sistemas seguros torna-se uma questão do gerenciamento de risco. Riscos de segurança devem ser identificados, classificados, e mitigados, e estes riscos devem ser gerenciados através do ciclo de vida do produto de software.

Riscos podem ser reduzidos pela seleção de vários controles ou riscos podem ser transferidos através de seguros. O gerenciamento de riscos utiliza uma combinação destes métodos.

4.1.4 Qualificar Processo e Práticas para Produzir Software Seguro

Para garantir que as práticas e processos futuros de software consistentemente produzam sistemas seguros, processos e práticas candidatos e os produtos que eles produzem devem ser analisados e testados para verificar e validar que os processos e práticas são confiáveis na produção de sistemas seguros.

A verificação de que o processo de software pode consistentemente produzir um sistema seguro é um desafio devido a, pelo menos, sete razões:

1. Segurança é um território de ameaças evolutivas. O que parece ser um sistema seguro hoje poderia ser inseguro no futuro.
2. Enquanto o rastreamento da habilidade do sistema em suportar ataques fornece alguma confiança de sua segurança, não há técnicas aceitas que possam provar sua segurança. Com os métodos correntes, pode-se apenas provar que o sistema não é seguro.
3. O número de defeitos de segurança toleráveis é pequeno e verificar que um número ainda menor de defeitos existe em grandes programas é extremamente difícil.
4. Mesmo se um sistema seguro foi inicialmente desenvolvido, a melhoria de sua entrega, seu reparo, e sua remediação não deve comprometer sua segurança.
5. Mesmo após um ou mais processos terem produzido um sistema seguro, esses processos e práticas devem se manter eficazes quando usados por muitas pessoas em muitas empresas de software e muitos ambientes de desenvolvimento.
6. Um esforço extensivo e estendido de coleção de dados seria requerido para obter significante evidência de que o processo produz sistema seguro.
7. Os métodos para qualificar as capacidades dos processos e empresas demorariam e seriam custosos.

4.1.5 Sugestão de Verificação e Estratégia de Qualificação

Para qualificar um processo como capaz de produzir sistema seguro, esse processo deve ser usado para desenvolver, melhorar, e/ou remediar múltiplos produtos de software e, então, esses produtos devem ser testados ou examinados para verificar que estão seguros. As estratégias de verificação sugeridas incluem:

- Desenvolvedores e mantenedores de produtos de software seguro utilizam processos e práticas que mostrem produzir produtos de software com pouco defeito.

- Para acompanhar os processos e práticas, desenvolvedores e mantenedores de software devem ser treinados, gerenciados, apoiados de tal forma que possam manter o nível exigido de disciplina.
- As equipes de desenvolvimento devem medir e gerenciar seus processos de software de tal forma que eles melhorem.
- Nos níveis de especificação e projeto, utilizar os melhores métodos disponíveis para contemplar as propriedades de segurança emergentes do software.
- As equipes de desenvolvimento devem rastrear e analisar todo defeito de segurança encontrado em todo produto gerado por cada equipe, a fim de entender porque e como aquele defeito foi inserido, onde defeitos similares ainda poderiam permanecer no produto, e como encontrar e corrigi-los de forma eficiente.
- Quando os produtos de segurança produzidos são utilizados, novas vulnerabilidades devem ser rastreadas com as versões do produto e os módulos onde elas foram encontradas. As práticas utilizadas para desenvolver esses módulos devem ser identificadas, e as falhas no processo que permitiram os defeitos serem inseridos e/ou esquecidos devem ser identificadas, e uma correção no processo deve ser desenvolvida e analisada para garantir que não haja impacto negativo. As equipes devem ajustar seus processos para prevenir e encontrar todos os defeitos parecidos.
- O processo inclui como os defeitos de segurança são corrigidos e a correção disseminada aos usuários do produto, envolvendo o defeito imediato e todos os outros de natureza similar que são identificados como parte do processo.
- Dados do processo devem ser agrupados e retidos sempre que esse processo for utilizado. Esses dados devem ser usados por revisores de processo e nomeados para qualificar os produtos. Os registros do resultado da qualificação devem incluir dados sobre os produtores, quem utilizou o processo, como eles utilizaram o processo, como eles foram treinados e preparados a usar o processo, e o ambiente no qual o processo foi usado. Os dados devem estar disponíveis para produzir e verificar a qualidade e segurança do produto e do treinamento e qualificação dos analistas, revisores, e testadores. Estes registros devem ser mantidos de tal forma que, se em um produto qualificado como seguro seja encontrado algum problema de segurança, ações apropriadas sejam tomadas.

- Se o processo foi utilizado indevidamente, o treinamento e a qualificação dos desenvolvedores e revisores designados deveriam ser avaliados e ajustados. Se o uso impróprio do processo foi um problema persistente, os registros dos desenvolvedores e revisores deveriam ser revisados e ações tomadas para revogar quaisquer qualificações não mais apropriadas. Alguma ação deve ser tomada para revisar o histórico do processo, para decidir se o processo poderia ser adequadamente corrigido ou ser desqualificado.

4.1.6 Recomendações Finais

Dado um processo que produza software de alta qualidade, a próxima recomendação é tornar a gerência de risco central à tomada de decisão. Isto requer experiência em segurança que cobre todos os aspectos de segurança do sistema sob desenvolvimento. Desde que a experiência de segurança seja ampla e profunda, ajuda especializada pode ser necessária para identificar e gerenciar riscos de segurança.

Segurança deve ser uma consideração integral durante especificação e projeto do produto. É importante assegurar também que as propriedades de segurança do software foram definidas, e que especificações e projetos para segurança sejam analisadas e revisadas. Um elemento chave para tornar isso possível é projetar o software para que os aspectos de segurança estejam concentrados em uma porção limitada do software. O projeto deve ser tão simples quanto possível – mesmo que sacrifique a eficiência – e deve ser restrito a estruturas e características que são "seguras" e, de preferência, possam ser analisadas. O projeto não deve assumir que o software não possa ser "quebrado", e deveria garantir defesa em profundidade ou tolerância.

4.2 GASSP

GASSP (*Generally Accepted System Security Principles*) relaciona-se com a segurança física, técnica e administrativa da informação, envolvendo princípios de segurança genéricos, funcionais, e detalhados (POORE, 1999).

Propõe que a otimização das práticas de segurança pode melhorar a eficácia e a eficiência do negócio, incrementando sua produtividade e aumentando o nível de confiança das informações processadas. Este melhoramento poderá permitir o desenvolvimento de produtos com padronização global, garantindo maior confiança, aceitação e satisfação dos clientes.

4.2.1 Princípios Genéricos

Os princípios genéricos fornecem guias gerais para estabelecer e manter a segurança da informação. Esses princípios formam a base dos princípios funcionais e detalhados.

4.2.2 Princípios Funcionais

Os princípios funcionais são derivados dos princípios genéricos que representam os objetivos conceituais da segurança da informação.

Por guiar a realização operacional dos princípios genéricos, os princípios funcionais definem "o que fazer" e permitem a definição das unidades básicas desses princípios.

Por apresentarem os princípios funcionais um menor escopo, eles são mais fáceis de tratar no que concerne o planejamento da implementação e execução.

Alguns princípios funcionais incluem:

- *Política de segurança da informação*: a gerência deve garantir que política e padrões, *baselines*, procedimentos e guias de apoio sejam desenvolvidos e mantidos para satisfazer aspectos de segurança da informação.
- *Responsabilização*: a gerência deve garantir a responsabilização das pessoas pelo acesso e uso da informação.
- *Gerenciamento da informação*: a gerência deve rotineiramente catalogar e valorizar os ativos de informação, e atribuir níveis de sensibilidade e criticidade.
- *Integridade do sistema*: a gerência deve garantir que as propriedades dos sistemas e aplicações, que são essenciais para apoiar a missão da organização, sejam estabelecidas, preservadas e protegidas.
- *Ciclo de vida dos sistemas de informação*: a gerência deve garantir que segurança é tratada em todos os estágios do ciclo de vida do sistema.
- *Segurança de rede e de infra-estrutura*: a gerência deve considerar o impacto potencial na infra-estrutura global e em outros sistemas conectados quando definir as medidas de segurança de rede.
- *Requisitos legais, regulatórios e contratuais de segurança da informação*: a gerência deve estar atenta e tratar todos os requisitos legais, regulatórios e contratuais relacionados aos ativos de informação.

4.2.3 Princípios Detalhados de Segurança

Os princípios detalhados de segurança tratam especificamente de métodos de obtenção de conformidade com os princípios funcionais, com respeito aos ambientes existentes e tecnologia disponível.

Há muitos princípios detalhados de segurança da informação apoiando um ou mais princípios funcionais. Os princípios detalhados lidam com tecnologias, ambientes, padrões, práticas, e conceitos diferenciados que são relevantes para os princípios funcionais. Os princípios detalhados evoluem continuamente, para superar os desafios das tecnologias emergentes e novas ameaças (POORE, 1999).

4.3 SPSMM

O SPSMM *(Secure Programming Standards Methodology Manual)* é um manual que faz parte do trabalho do OSSTMM (*Open Source Security Testing Methodology Manual*) (RODRIGUEZ, 2002).

Foi elaborado para programadores que desejam estabelecer guias e/ou procedimentos para programação segura em projetos de tecnologia da informação, independente da linguagem de programação, ambiente da aplicação e ferramentas de desenvolvimento.

Define parâmetros e tarefas orientadas para programação segura, como os que se seguem:

- *Entrada de dados*: não é apenas realizada pelo usuário ou interface de programa, mas também aquela obtida de um sistema e outro aplicativo que esteja em conformidade com a base do sistema. Uma vez que a proteção tenha sido estabelecida para o mundo externo (entrada de dados), devemos proteger nossas próprias práticas internas e de codificação.
- *Consumo de recursos*: todo programa consome recursos para seus próprios propósitos. Dentro de certas condições, é possível que o programa comece um ciclo de consumo que termine em instabilidade ou o programa corrompido. Esses problemas causam o que se conhece como "Negação de serviço" (DoS – *Denial of Service*). É uma boa prática limitar a quantidade de recursos usados, e rejeitar pedidos quando um nível especificado de consumo de recursos for alcançado.
- *Saída de dados*: erros de saída podem ser vistos como erros de processo, porque produzem um produto com uma saída inesperada. O que se tentará representar são informações passando nos canais normais de saída, como: erros padrões que possam divulgar informação de estrutura interna, problemas no estilo, etc.

- *Níveis de permissão dos dados*: existem algumas categorias (ultra-secreto, secreto, público, etc.) que classificam cada objeto (dado, etc.) e uma pessoa que possa acessar qualquer recurso classificado com a mesma categoria ou com categoria menor que a sua. Por exemplo: não se pode transmitir um documento "ultra-secreto" através de um canal "secreto", pois o canal tem mecanismos de proteção menores que o documento transmitido pode aceitar.

4.4 Melhores Práticas para Desenvolvimento

Peteanu (2001) desenvolveu uma pesquisa, levantando melhores práticas para o desenvolvimento de aplicações seguras. Isto envolve a segurança no ciclo de vida do projeto, e alguns princípios para o desenvolvimento seguro.

4.4.1 Segurança no Ciclo de Vida do Projeto

Um projeto para contribuir com a melhoria da segurança de software não deve focar apenas em codificação. Além disso, construir segurança dentro da aplicação não se inicia com a implementação das tecnologias de segurança. Não obstante, construir segurança envolve desde a identificação das necessidades até a realização de teste de segurança. A segurança no projeto se dá em quatro fases:

- *Concepção*: define o escopo do produto. A segurança geralmente ainda não é um tópico de discussão. Todavia, pode ser útil reconhecer a importância da segurança nos estágios subseqüentes e incluir um membro na equipe que seja responsável pelos aspectos de segurança.
- *Elaboração*: identifica ativos, realiza a análise de ameaça. A arquitetura do sistema deve fornecer um entendimento claro de como a segurança é fornecida pelo sistema e dentro dele, como confiança é gerenciada e como o sistema resiste aos maiores tipos de ataque.
- *Construção*: ocorre o projeto e implementação detalhados do sistema. Todas as condições de ataque podem ser testadas, mas é necessário realizar testes relevantes de segurança. Isto requer que os testadores tenham habilidades e conhecimento para atacar sistemas, o que não representa a realidade.
- *Transição*: o software é entregue aos usuários e passa a ser mantido. Deve-se ter certeza de que a documentação técnica descreva os aspectos de segurança associados ao sistema, incluindo a documentação legal.

4.4.2 Princípios para o Desenvolvimento Seguro

Os princípios para o desenvolvimento seguro são apresentados a seguir:

- *Relevância e lado mais fraco*: soluções de segurança são idealizadas para proteger um sistema ou dado de uso e divulgação não autorizados. Isso significa garantir que os reais problemas sejam tratados, e não colocar segurança onde não solucione um problema. O primeiro foca no lado mais fraco, pois tal lado mais fraco seria o primeiro lugar onde o atacante agiria.
- *Clareza*: uma solução de segurança deve ser clara em termos de tecnologia e processos. Ser vago significa que o comportamento do sistema não é compreensível, nem mesmo pelas pessoas que o construíram.
- *Qualidade*: segurança é mais uma qualidade do sistema do que uma questão de características. Nesse sentido, características de segurança só aumentariam a segurança do sistema, se apropriadamente implementadas e se forem relevantes em relação aos riscos identificados.
- *Participação de todos os envolvidos*: os sistemas e sua segurança são importantes não apenas para a equipe de desenvolvimento, mas também para a equipe de infra-estrutura e para as áreas de negócio – é necessário envolver todos.
- *Operação de falha segura*: sistemas devem ter um comportamento que falhe de forma segura. Se o subsistema de segurança torna-se não operacional, deveria levar a um estado onde segurança não é afetada.
- *Defesa em profundidade*: se há apenas um mecanismo de segurança, em algum dia esse mecanismo poderá falhar, expondo o sistema por completo. É necessário projetar camadas de mecanismos de segurança de tal forma que, se um deles falhe, o dano seja reduzido.
- *Privilégios*:
  - *Menor privilégio*: só conceda o mínimo de privilégios necessários para realizar uma operação pelo menor tempo.
  - *Privilégio necessário*: é preciso analisar quais privilégios são necessários em determinados momentos.
- *Interação com usuários*: algumas verdades são válidas em relação à interação com os usuários:
  - não “confie” nos usuários;

  - não aborreça os usuários, perguntando-lhes inútil e constantemente por informação segura;
  - não permita que usuários forneçam senhas fracas;
  - escolha a configuração padrão como a mais segura;
  - esconder aspectos de segurança para o usuário, interferindo o mínimo na usabilidade do sistema.
- *Limites de confiança*: um sistema complexo poderá ter camadas e subsistemas. Uma arquitetura segura deveria capturar quais destas entidades confiam em quem e porque.
- *Software de terceiros*: uma característica desejável é a habilidade de isolar a parte vulnerável do software de terceiros e/ou monitorar o acesso a ele de modo que seja pelo menos notificado, quando uma invasão é tentada.
- *Ataque primeiro no papel*: durante o projeto da arquitetura de segurança, tente quebrá-la. É quase certo haver problemas a sanar após os primeiros ataques. Se nada for encontrado, tente novamente e peça a outros que façam o mesmo. Esteja informado sobre ataques contra esquemas similares e preste atenção no método de ataque.
- *Declarativo versus programático*: se as restrições de segurança são cumpridas durante a execução do programa, há duas abordagens: o próprio programa checa as condições e toma a decisão (programático) ou o ambiente a ser executado realiza as checagens fora do controle do programa (declarativo).
- *Revisões são boas práticas*: é importante que haja revisões de projeto e revisões de código para assegurar um produto e um processo seguro.

4.5 O Processo SDL

O processo SDL – *Trustworthy Computing Security Development Lifecycle* – elaborado pela Microsoft é composto por um conjunto de medidas de segurança a serem inseridas durante o ciclo de desenvolvimento (HOWARD, 2002).

Este processo assume que haja um grupo na organização que guie o desenvolvimento e a evolução de melhores práticas de segurança e melhorias do processo, servindo como fonte de conhecimento para a organização e que realize uma revisão de segurança antes da liberação do sistema. Essa equipe de segurança deve interagir freqüentemente durante projeto e desenvolvimento de software.

4.5.1 Visão Geral do SDL

A experiência da Microsoft com a segurança de sistemas levou a definição de um conjunto de princípios para construir software mais seguro, que foi chamado de SD3+C. Esses princípios são: (HOWARD, 2002):

- *Seguro por projeto*: o software deve ser arquitetado, projetado, e implementado de forma a se proteger e também proteger a informação processada, bem como resistir a ataques.
- *Seguro por padrão*: o software nunca atingirá a segurança perfeita; então, projetistas devem assumir que falhas de segurança estejam presentes. Para minimizar o dano ocorrido quando atacantes exploram vulnerabilidades remanescentes, o estado padrão do software deve ser seguro. (McGRAW, 2000e).
- *Seguro na entrega*: ferramentas e guias deveriam acompanhar o software para ajudar os usuários finais e/ou administradores a usá-lo seguramente.
- *Comunicações*: os desenvolvedores devem estar preparados para descobrir vulnerabilidades nos produtos e comunicar-se com usuários finais e/ou administradores, para ajudá-los a tomar ações de proteção.

As medidas de segurança no paradigma SD3+C no processo de desenvolvimento são apresentadas na Figura 4.2, envolvendo seis fases.

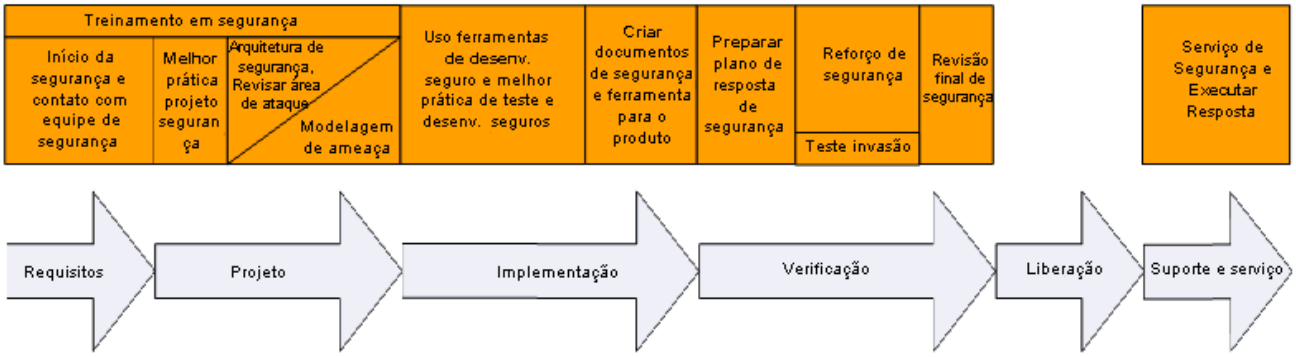

Figura 4.2: SDL (HOWARD, 2002)

4.5.2 Fases do SDL

As fases do SDL são: requisitos, projeto, implementação, verificação, liberação, e suporte e serviço.

Requisitos

Na fase de requisitos, a equipe responsável pelo produto contata a equipe de segurança para solicitar a participação do consultor de segurança que serve como ponto de contato,

recurso e guia enquanto o planejamento prossegue. O consultor de segurança auxilia a equipe do produto revisando planos, fazendo recomendações, e garantindo que a equipe de segurança planeje recursos adequados para apoiar o processo e o cronograma da equipe do produto. O consultor de segurança explica à equipe do produto sobre os marcos de segurança e critérios de saída que serão necessários baseados no tamanho do projeto, complexidade, e risco.

O consultor de segurança permanece como ponto de contato da equipe do produto com a equipe de segurança desde a concepção do projeto até o término da revisão final de segurança e liberação do sistema. O consultor também é ponto de contato entre as gerências das duas equipes, avisando-as se os elementos de segurança do projeto estão sendo seguidos a fim de evitar surpresas de segurança durante o processo.

A fase de requisitos é a oportunidade para a equipe do produto considerar como a segurança será integrada no processo de desenvolvimento, identificar objetivos chave de segurança, e maximizar a segurança do software enquanto minimiza problemas nos planos e cronogramas.

Como parte deste processo, a equipe do produto necessita considerar como os atributos de segurança e medidas de garantia do produto integrarão outros produtos para ser usado conjuntamente. A perspectiva geral da equipe do produto sobre os objetivos de segurança, desafios, e planos deve ser refletida nos documentos de planejamento produzidos durante esta fase.

Projeto

Em uma perspectiva de segurança, os elementos chaves da fase de projeto são:

- *Definir arquitetura de segurança e guias de projeto*: estrutura geral do software em uma perspectiva de segurança, e identificar os componentes cujo funcionamento correto seja essencial à segurança.
- *Documentar os elementos da área de ataque do sistema*: é importante que apenas as funcionalidades a serem usadas pela maioria dos usuários sejam expostas a todos os usuários por padrão, e que tais funcionalidades sejam instaladas com o menor nível adequado de privilégio.
- *Conduzir modelagem de ameaças*: a equipe do produto identifica os ativos que o software deve gerenciar e as interfaces pelas quais esses ativos podem ser acessados. A equipe deve identificar proteções que reduzem o risco.

- *Definir critérios suplementares de entrega*: enquanto critérios básicos de entrega segura devem ser definidos em nível organizacional, equipes do produto podem ter critérios específicos que devem ser obedecidos antes da liberação do produto.

Implementação

Durante a fase de implementação, a equipe do produto codifica, testa, e integra o software. Passos realizados para remover falhas de segurança ou prevenir sua inserção durante esta fase são influenciados. Isto reduz a probabilidade de que vulnerabilidades de segurança estejam presentes na versão final do software entregue aos clientes.

Os elementos que se aplicam à fase de implementação são:

- *Aplicar padrões de codificação e teste*: padrões de teste e melhores práticas ajudam a garantir que o teste seja focado na detecção de vulnerabilidades potenciais de segurança em vez de concentrar apenas na operação correta das funcionalidades e características do software.
- *Aplicar ferramentas de teste de segurança*: ferramentas fornecem entradas estruturadas, maximizando a probabilidade de detectar erros que possam ocasionar vulnerabilidades de software.
- *Aplicar ferramentas de análise estática do código*: ferramentas podem detectar alguns tipos de falhas de codificação, que resultam em vulnerabilidades, incluindo sobrecarga de memória e variáveis não inicializadas.
- *Conduzir revisões de código*: devem ser conduzidas por desenvolvedores treinados em examinar código fonte, detectar e remover vulnerabilidades potenciais de segurança.

Verificação

Durante a fase de verificação, a equipe do produto produz um "reforço de segurança", que inclui revisões de segurança do código, além das realizadas na fase de implementação.

O "reforço de segurança" é uma oportunidade a mais para revisar o código que foi desenvolvido ou atualizado durante a fase de implementação e o código legado que não sofreu alteração.

É importante ressaltar que as revisões de código e teste do código prioritário (código que faz parte da "área de ataque" do software) são críticos para várias partes do SDL.

Liberação

Durante a fase de liberação, o software deve passar por uma Revisão Final de Segurança (*Final Security Review* - FSR), que objetiva responder à questão: "*Do ponto de vista da segurança, este software está pronto para ser colocado em produção*?"

A FSR pode ser conduzida de dois a seis meses antes de o software ficar completo, dependendo de seu escopo. É uma revisão independente do software conduzida pela equipe de segurança da organização.

A FSR inicia pelo preenchimento de um questionário realizado pela equipe do produto, e uma entrevista com um membro da equipe de segurança responsável pelo FSR. Qualquer FSR necessitará fazer uma revisão das falhas (*bugs*) identificadas em um primeiro momento como falhas de segurança, mas que depois não impactariam na segurança, para assegurar que a análise foi feita corretamente. Também inclui uma revisão da habilidade do software em lidar com novas vulnerabilidades reportadas que afetam produtos de software similares. Para sistemas de grande porte, esta revisão pode necessitar de testes de invasão e, potencialmente, a contratação de revisão externa de segurança.

A FSR dá à equipe do produto e aos altos executivos uma visão geral da postura de segurança do sistema e sua probabilidade de superar ataques após entrega aos clientes. Se esta revisão encontrar um padrão de vulnerabilidades remanescente, procura não apenas corrigir essas vulnerabilidades, como também rever a fase anterior e realizar ações para tratar a origem do problema.

Suporte e Serviço

Apesar da aplicação do SDL durante o desenvolvimento, práticas maduras de desenvolvimento ainda não garantem a entrega de software sem vulnerabilidades.

Mesmo se o processo de desenvolvimento pudesse eliminar todas as vulnerabilidades do software a ser entregue, novos ataques poderiam ser descobertos e o software considerado seguro poderia ficar vulnerável. Equipes do produto devem estar preparadas para responder a novas vulnerabilidades descobertas no software antes da entrega ao cliente.

Parte do processo de resposta envolve a preparação para avaliar relatórios de vulnerabilidades e liberar atualizações e alertas de segurança, quando apropriado. Outro componente do processo de resposta é conduzir análise *post-mortem* das vulnerabilidades reportadas e tomar medidas quando necessário. Ações em resposta a uma vulnerabilidade variam da atualização em resposta a um erro isolado à atualização de ferramentas de análise de código.

O objetivo durante a fase de resposta é aprender com os erros e usar a informação fornecida nos relatórios de vulnerabilidades, para ajudar a detectar e eliminar vulnerabilidades antes de serem descobertas em produção e que possam colocar o cliente em risco. O processo de resposta também ajuda as equipes do produto e de segurança a adaptarem processos para que erros similares não sejam introduzidos no futuro.

4.5.3 A Utilização do SDL

A experiência da Microsoft indica que o SDL é eficaz na redução da incidência de vulnerabilidades de segurança. A implementação inicial do SDL resultou em melhorias significantes na segurança de um *software* específico, e subseqüentes versões desse *software*, refletindo em melhorias no SDL (LIPNER, 2005).

Implementação incremental dos elementos que compõem o SDL tem ocasionado melhorias incrementais, as quais são um sinal de um processo eficaz. Contudo, o processo não é perfeito e ainda evolui.

4.6 OWASP

O OWASP (*Open Web Application Security Project*) (CURPHEY, 2002) é uma referência aberta para arquitetos, desenvolvedores, vendedores, consumidores, e profissionais de segurança envolvidos em projeto, desenvolvimento, e teste da segurança de aplicações *Web* e *Web Services*, objetivando auxiliar na construção de aplicações e serviços *Web* mais seguros.

O OWASP descreve componentes técnicos, personagens, processos, e questões gerenciais necessários para projetar, construir e manter uma aplicação *Web* segura. É um modelo de apoio capaz de ser adaptado durante as fases de um ciclo de vida. A idéia é sempre determinar primeiro qual o nível apropriado de segurança para a aplicação funcionar como planejado em seu ambiente próprio.

O OWASP apresenta o seguinte conjunto de guias de segurança:

- Validar entrada e saída de dados.
- Falhar de forma segura.
- Manter-se simples.
- Utilizar e reutilizar componentes confiáveis.
- Defesa em profundidade.
- Segurança por obscuridade não funciona.
- Menor privilégio.

- Separação de privilégios.
- Não transmitir senhas em aberto.
- Configuração padrão segura.

Além disso, o projeto OWASP aborda a segurança da informação nos seguintes assuntos: arquitetura, autenticação, gerência de sessões de usuários, controle de acesso e autorização, registro de eventos, validação de dados, prevenção de problemas, privacidade, e criptografia.

4.7 Conclusão

Este capítulo apresentou um conjunto de trabalhos relacionados com segurança de software. Apesar de fornecer idéias ou informações úteis em contextos específicos de segurança de software, esses trabalhos, ou boas práticas de mercado, em geral, devem ser aplicados com cuidado em um contexto de processo. Isso se deve à ausência de consistência e de provas formais, na maioria dessas boas práticas, de que sua aplicação realmente produz resultados satisfatórios, com garantia de retorno dos recursos investidos.

No próximo capítulo, será apresentado o resultado de uma pesquisa de campo.

Capítulo **5**

Importância das Atividades de Segurança no Processo de Apoio à Segurança de Software

Este capítulo apresenta e analisa os resultados de uma pesquisa de campo feita com especialistas em processo de desenvolvimento de software e especialistas em segurança da informação.

Este trabalho propôs um conjunto inicial de atividades para um processo para apoiar o desenvolvimento de software mais seguro (processo seguro) a partir do SSE-CMM (2003), do OCTAVE (ALBERTS, 2001), da ISO/IEC 27002 (2005) e da ISO/IEC 15408 (2005a, 2005b, 2005c), o qual foi avaliado através de uma pesquisa de campo por 42 especialistas em processo de desenvolvimento de software e especialistas em segurança da informação de várias instituições públicas e privadas distribuídas entre as regiões do Brasil. Com os resultados obtidos nessa pesquisa de campo, as atividades avaliadas foram refinadas e geraram as atividades do Processo de Apoio à Segurança de Software proposto no capítulo 6.

A condução desta pesquisa de campo ocorreu a partir da aplicação de um questionário baseado em Andrade (2005), Apêndice 2, em que foi identificado o grau de importância das 40 atividades inicialmente propostas para o Processo de Apoio à Segurança de Software. Dos questionários aplicados foram descartados apenas três deles, por terem sido preenchidos de forma incompleta.

Esta pesquisa de campo foi divida em três blocos. O primeiro bloco traçou o perfil dos especialistas em:

- Segurança da informação:
    - *Como você classificaria seu conhecimento na área de segurança da informação*?
    - *Como você classificaria sua experiência prática em segurança da informação*?

- Processo de software:
    - *Como você classificaria seu conhecimento na área de processo de software*?
    - *Como você classificaria sua experiência prática em processo de software*?

O segundo bloco engloba questões relativas ao grau de importância de cada atividade desse processo seguro, a partir da escala abaixo:

- 0: atividade não é importante para o processo seguro.
- 1: atividade pouco importante para o processo seguro.
- 2: atividade com média importância para o processo seguro.
- 3: atividade importante para o processo seguro.
- 4: atividade muito importante para o processo seguro.

O terceiro bloco elenca um conjunto de aspectos que podem influenciar a utilização de um processo seguro a partir da escala abaixo:

- 0: aspecto não é influente na utilização do processo seguro.
- 1: aspecto com pouca influência na utilização do processo seguro.
- 2: aspecto com média influência na utilização do processo seguro.
- 3: aspecto influente na utilização do processo seguro.
- 4: aspecto com muita influência na utilização do processo seguro.

Os resultados obtidos nesta pesquisa serão apresentados, enfocando a: (*i*) avaliação consolidada dos especialistas em relação à importância das atividades do processo seguro; (*ii*) avaliação dos especialistas em processo de software em relação à importância das atividades do processo seguro; (*iii*) avaliação dos especialistas em segurança da informação em relação à importância das atividades do processo seguro; e (*iv*) avaliação consolidada dos especialistas sobre os aspectos que podem influenciar a utilização de um processo seguro.

5.1 Avaliação Consolidada dos Especialistas

A Tabela 5.1 mostra, em ordem decrescente do grau de importância (média aritmética e desvio padrão), todas as atividades analisadas pelos especialistas desta pesquisa de campo.

A Figura 5.1 mostra, em ordem decrescente do grau de importância, as sete atividades melhor avaliadas do processo seguro proposto, por todos os especialistas desta pesquisa. Elas representam as atividades que obtiveram avaliações iguais ou superiores a 3,50. Nas demais representações procurou-se seguir o padrão de 7 atividades para aquelas melhor avaliadas.

A atividade "Definir objetivos de planejamento de segurança e identificar seus mecanismos" foi a melhor avaliada com o valor de 3,72, tendendo para muito importante em um processo seguro. Isto evidencia a relevância do planejamento da segurança, através da definição de objetivos de segurança e da identificação de mecanismos necessários e suficientes para o gerenciamento do projeto.

Tabela 5.1: Atividades do processo seguro avaliadas pelos especialistas

Atividades do Processo Seguro	Média	Desvio padrão
Definir objetivos de planejamento de segurança e identificar seus mecanismos	3,72	0,76
Analisar as vulnerabilidades de segurança identificadas	3,67	0,53
Identificar as ameaças de segurança aos ativos críticos	3,64	0,58
Definir requisitos de segurança	3,59	0,59
Executar métodos de identificação de vulnerabilidade de segurança	3,54	0,55
Compreender as necessidades de segurança do cliente	3,54	0,64
Realizar verificação de segurança	3,51	0,64
Atribuir responsabilidades de segurança no projeto	3,51	0,82
Realizar validação de segurança	3,49	0,64
Realizar auditorias de segurança	3,38	0,81
Planejar o gerenciamento de incidentes de segurança	3,36	0,78
Gerenciar a implementação de controles de segurança	3,31	0,69
Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, no ambiente, e em suas características	3,31	0,77
Priorizar riscos de segurança	3,28	0,69
Identificar e revisar requisitos de garantia de segurança	3,28	0,76
Obter acordo sobre requisitos de segurança	3,26	0,85
Avaliar risco de exposição de segurança	3,26	0,88
Gerenciar percepção, treinamento e programa de educação de segurança	3,23	0,78
Classificar as ameaças de segurança aos ativos	3,23	0,81
Entender e revisar necessidades de informação de segurança	3,23	0,84
Desenvolver estratégias de redução das ameaças de segurança	3,15	0,87
Priorizar processos críticos influenciados pelo sistema	3,13	0,83
Reavaliar mudanças nas ameaças, vulnerabilidades, impactos, riscos e no ambiente	3,13	0,95
Identificar e preparar a resposta dos incidentes de segurança relevantes	3,10	0,75
Definir estratégia de manutenção da garantia de segurança	3,10	0,88
Revisar e comunicar resultados de verificação e validação de segurança	3,05	0,83
Analisar registro de evento com impacto na segurança	3,05	0,83
Conduzir análise de impacto de segurança das mudanças	3,05	0,86
Fornecer orientação de segurança	3,03	0,78
Definir a abordagem de verificação e validação de segurança	3,00	0,79
Identificar exposição de segurança	3,00	0,86
Revisar o comportamento de segurança do sistema para identificar mudanças necessárias	3,00	0,89

Identificar e descrever impactos de segurança	2,97	0,81
Identificar e analisar alternativas de segurança	2,95	0,72
Gerenciar e controlar serviços e componentes operacionais de segurança	2,90	0,82
Capturar uma visão de alto nível orientada à segurança da operação do sistema	2,90	0,97
Determinar considerações e restrições de segurança	2,85	0,78
Revisar ativos do sistema que se referem à segurança	2,85	0,96
Controlar as evidências da manutenção da garantia de segurança	2,79	0,89
Implementar ambientes de processamento	2,74	0,79

A atividade "Analisar as vulnerabilidades de segurança identificadas" foi a segunda avaliada com o valor de 3,67, convergindo para o patamar de *muito importante* em um processo seguro. Neste contexto, é altamente recomendável que seja realizada uma avaliação, que determine se as vulnerabilidades de segurança identificadas possam levar à violação de funcionalidades de segurança por seus usuários.

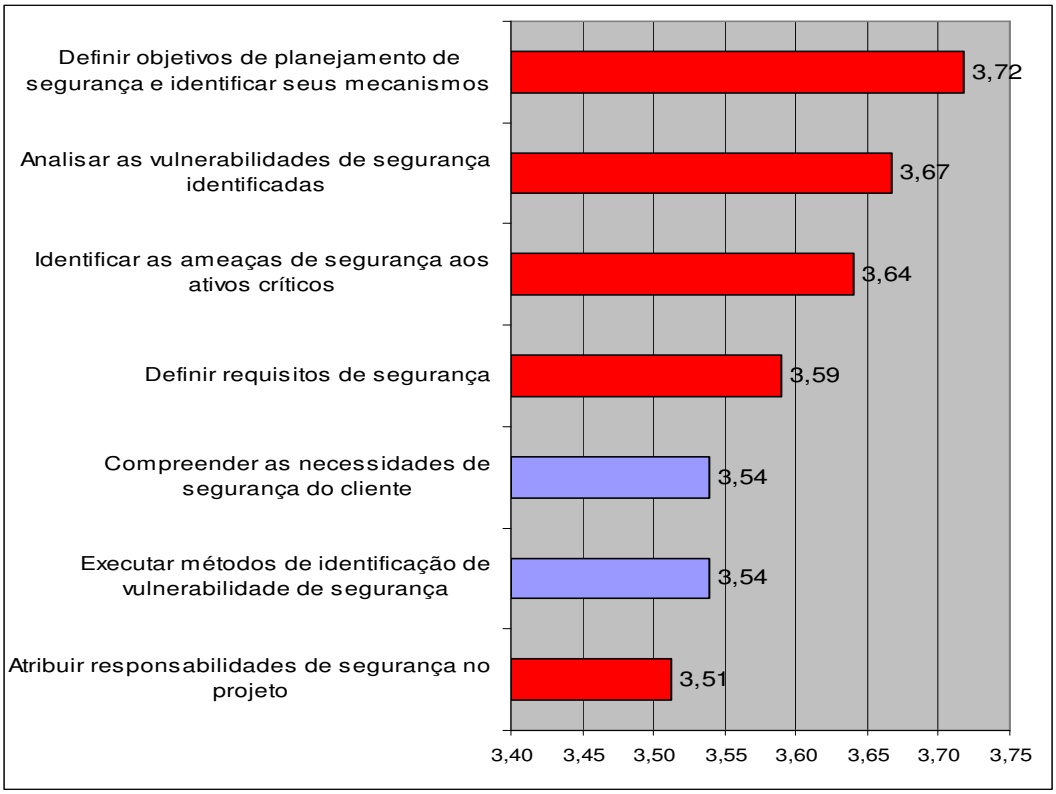

Figura 5.1 As sete atividades melhor avaliadas para o processo seguro

A atividade "Identificar as ameaças de segurança aos ativos críticos" foi a terceira melhor avaliada com o valor de 3,64, próximo do valor de *muito importante* estabelecido em um processo seguro. Isto reforça a importância de que as ameaças de segurança de cada ativo crítico do software devam ser identificadas, através de um mapeamento das áreas de perigo para cada ativo crítico de acordo com o perfil de ameaça daquele ativo.

A atividade "Definir requisitos de segurança" foi a quarta melhor avaliada com o valor de 3,59, posicionando-se entre a escala de valor *importante* e valor *muito importante* para um processo seguro. Isto ressalta a necessidade de definirmos um conjunto consistente de requisitos, que estabeleçam o nível de segurança a ser implementado em um sistema de informação.

As atividades "Compreender as necessidades de segurança do cliente" e "Executar métodos de identificação de vulnerabilidade de segurança" foram bem pontuadas e obtiveram um valor de 3,54. Isto mostra que todas as informações necessárias para um entendimento das necessidades de segurança do cliente devem ser coletadas. Da mesma forma, métodos, técnicas, e critérios devem ser executados para identificar e caracterizar as vulnerabilidades de segurança do software em um ambiente definido.

A atividade "Atribuir responsabilidades de segurança no projeto" obteve o valor de 3,51, evidenciando sua posição entre *importante* e *muito importante*. Portanto, as responsabilidades e as obrigações legais pelo cumprimento e gerenciamento de processos de segurança e pelos controles de segurança em um projeto devem ser atribuídas e comunicadas a todos os envolvidos na organização.

Comparando-se os resultados das avaliações realizadas por especialistas de processo de desenvolvimento de software e especialistas em segurança, é percebido que cinco das sete atividades melhor pontuadas são as mesmas tanto na avaliação dos especialistas em processo de software quanto na avaliação dos especialistas em segurança da informação, destacado em vermelho na Figura 5.1. Essas atividades de segurança foram: "Definir objetivos de planejamento de segurança e identificar seus mecanismos", "Analisar as vulnerabilidades de segurança identificadas", "Identificar as ameaças de segurança aos ativos críticos", "Definir requisitos de segurança", e "Compreender as necessidades de segurança do cliente". Essas cinco atividades obtiveram um valor maior ou igual a 3,55. Portanto, essas atividades foram consideradas de muita importância na estruturação de um Processo de Apoio à Segurança de Software.

A Figura 5.2 apresenta as três atividades que obtiveram os menores valores, em relação ao seu grau de importância, na avaliação de um processo seguro, segundo os especialistas desta pesquisa. Elas representam as atividades que obtiveram avaliações iguais ou menores do que 2,85. Nas demais representações procurou-se seguir o padrão de 3 atividades para aquelas pior avaliadas.

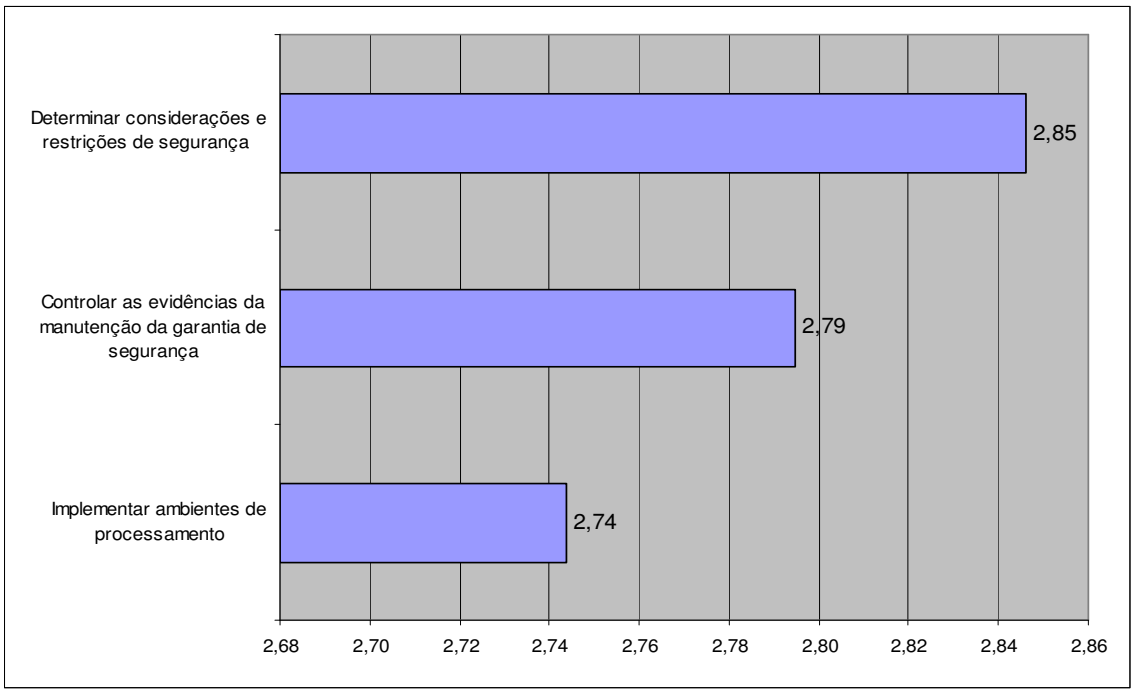

Figura 5.2. As três atividades de menor grau de avaliação do processo seguro

A atividade "Implementar ambientes de processamento" foi considerada de *menor importância* em processo para apoiar à segurança de software, obtendo o valor de 2,74. Isto significa que os especialistas deram importância mediana, tendendo para importante, à questão da implementação de ambientes de desenvolvimento, ambientes de ensaio e ambientes de produção, e à avaliação do nível de separação necessário entre eles para prevenir problemas operacionais.

As atividades "Controlar as evidências da manutenção da garantia de segurança" e "Determinar considerações e restrições de segurança" também tiveram uma avaliação de *importância mediana*, contudo mais próximas de *importante*, com valores de 2,79 e 2,85, respectivamente. Assim sendo, é considerada de alguma relevância a manutenção da confiança na garantia de segurança através de evidência, que ratifiquem a segurança do software. Isto pode auxiliar na tomada de decisões do processo e na realização de escolhas conscientes de itens de segurança da informação.

Em relação às atividades com a menor pontuação, apenas uma atividade foi comum em ambas as avaliações dos especialistas. Isto aconteceu com a atividade "Implementar ambientes de processamento" com valores de 2,77 e 2,70, apesar de ficar na faixa de valores de importância mediana.

5.2 Avaliação dos Especialistas em Processo de Desenvolvimento de Software

A Figura 5.3 mostra, em ordem decrescente de seu grau de importância, as sete atividades do processo seguro proposto melhor avaliadas pelos especialistas em processos de software.

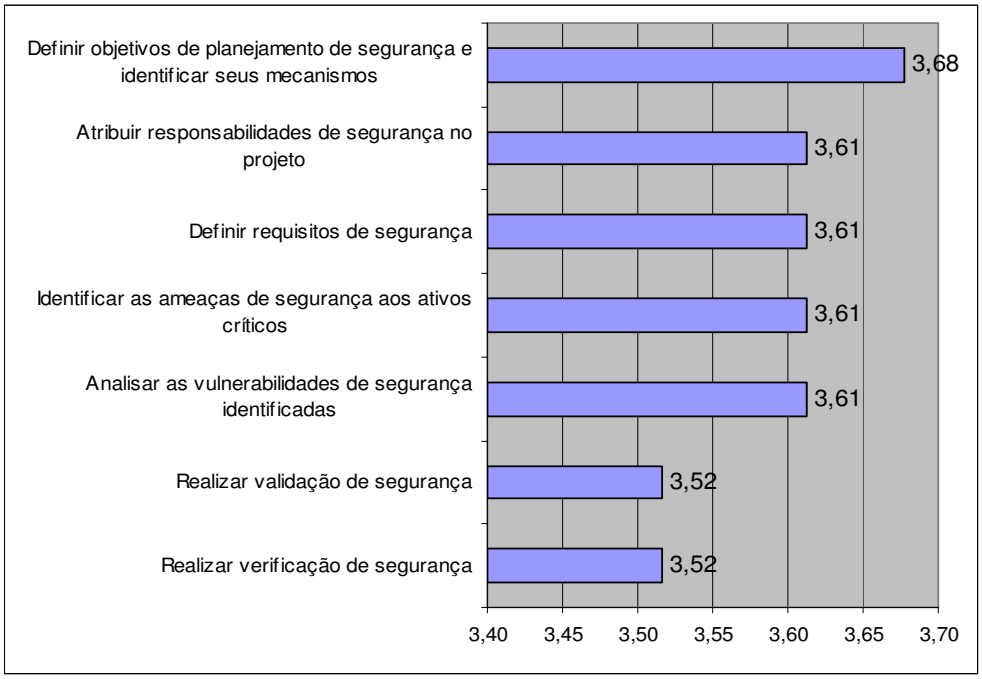

Figura 5.3 As atividades melhor avaliadas por especialistas em processos de software

A atividade "Definir objetivos de planejamento de segurança e identificar seus mecanismos" foi a melhor avaliada com o valor de 3,68, tendendo para o valor máximo da escala de avaliação. Isto evidencia a relevância do planejamento da segurança, por meio da definição de objetivos de segurança e da identificação de mecanismos necessários e suficientes para o gerenciamento do projeto.

As atividades "Atribuir responsabilidades de segurança no projeto", "Definir requisitos de segurança", "Identificar as ameaças de segurança aos ativos críticos", e "Analisar as vulnerabilidades de segurança identificadas" foram todas avaliadas com o valor de 3,61, convergindo também para o patamar de *muito importante* em um processo seguro.

Neste contexto, é altamente recomendável que seja realizada: (*i*) a atribuição e comunicação a todos os envolvidos das responsabilidades e das obrigações legais pelo cumprimento e gerenciamento de processos de segurança e pelos controles de segurança em um projeto; (*ii*) a definição de um conjunto consistente de requisitos, que estabeleçam o nível de segurança a ser implementado em um sistema de informação; (*iii*) a identificação das ameaças de segurança de cada ativo crítico do software, através de um mapeamento das áreas

de perigo para cada ativo crítico de acordo com o perfil de ameaça daquele ativo; e (*iv*) a avaliação, que determine se as vulnerabilidades de segurança identificadas possam levar à violação de funcionalidades de segurança por seus usuários.

As atividades "Realizar validação de segurança" e "Realizar verificação de segurança" foram bem pontuadas e obtiveram um valor de 3,52. Isto mostra que um processo de verificação e validação de segurança deve existir para garantir que a solução satisfaça às necessidades de segurança do cliente e que a solução implemente seus requisitos de segurança, respectivamente.

A Figura 5.4 apresenta as três atividades que obtiveram os menores valores, em relação ao seu grau de importância, na avaliação de um processo seguro, segundo os especialistas em processo de software.

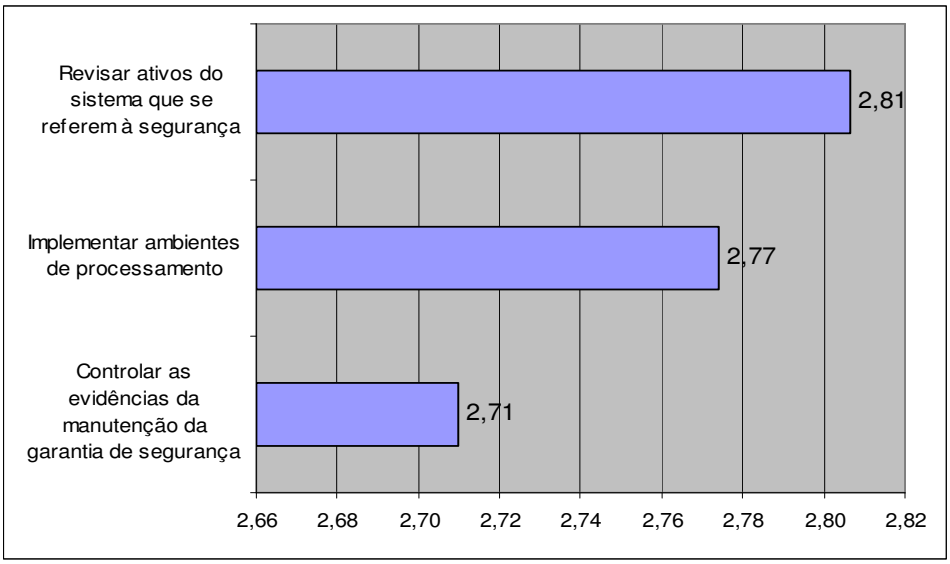

Figura 5.4 As atividades de menor grau de avaliação por especialistas em processos de software

A atividade "Controlar as evidências da manutenção da garantia de segurança" foi a considerada de menor importância em processo seguro, obtendo o valor de 2,71, em que os especialistas em processo de software deram importância mediana à questão do controle das evidências responsáveis pela manutenção da garantia de segurança.

As atividades "Implementar ambientes de processamento" e "Revisar ativos do sistema que se referem à segurança" também tiveram uma avaliação de importância mediana, contudo mais próximas de importante, com valores de 2,77 e 2,81, respectivamente. Assim sendo, é considerada de razoável importância a implementação de ambientes de desenvolvimento, ambientes de ensaio e ambientes de produção, e a avaliação do nível de separação necessário entre estes para prevenir problemas operacionais. É também considerada

de *média importância*, com tendência para importante, a revisão dos ativos do software que suportam as capacidades chaves ou objetivos de segurança.

5.3 Avaliação dos Especialistas em Segurança da Informação

A Figura 5.5 apresenta, em ordem decrescente do grau de importância, as sete atividades do processo seguro proposto melhor avaliadas pelos especialistas em segurança da informação.

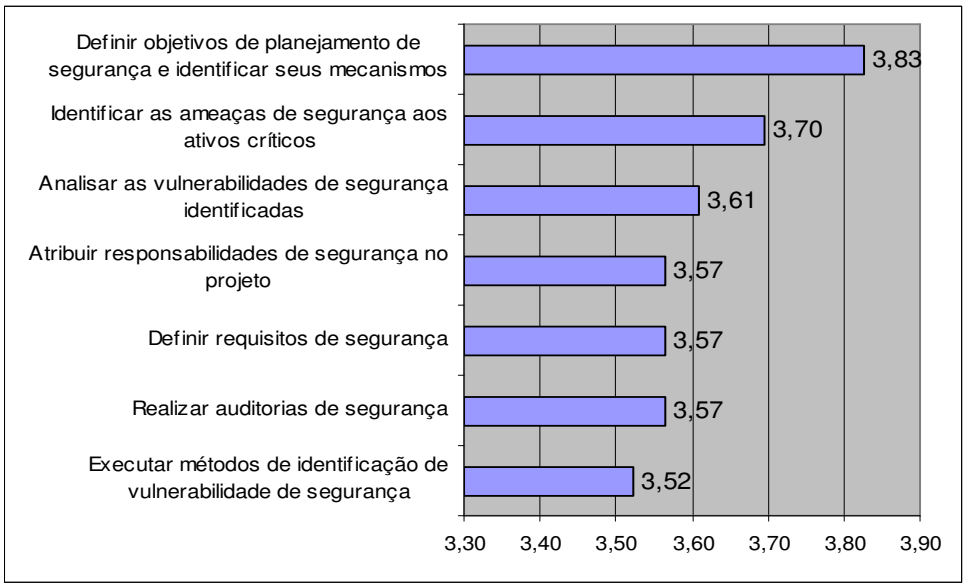

Figura 5.5 As atividades melhor avaliadas por especialistas em segurança da informação

A atividade "Definir objetivos de planejamento de segurança e identificar seus mecanismos" foi também a melhor avaliada pelos especialistas em segurança com o valor de 3,83, tendendo para muito importante em um processo seguro. Isto evidencia a relevância do planejamento da segurança, através da definição de objetivos de segurança e da identificação de mecanismos necessários e suficientes para o gerenciamento do projeto.

A atividade "Identificar as ameaças de segurança aos ativos críticos" foi a segunda melhor avaliada com o valor de 3,70, convergindo também para o patamar de muito importante em um processo seguro. Isto reforça a importância de que as ameaças de segurança de cada ativo crítico do software devam ser identificadas, através de um mapeamento das áreas de perigo, para cada ativo crítico de acordo com o perfil de ameaça daquele ativo.

A atividade "Analisar as vulnerabilidades de segurança identificadas" foi a terceira melhor avaliada com o valor de 3,61, próximo do valor de *muito importante* estabelecido em um processo seguro. Neste contexto, é altamente recomendável que seja realizada uma

avaliação, que determine se as vulnerabilidades de segurança identificadas possam levar à violação de funcionalidades de segurança por seus usuários.

As atividades "Atribuir responsabilidades de segurança no projeto", "Definir requisitos de segurança", e "Realizar auditorias de segurança" foram todas avaliadas com o valor de 3,57, sendo muito bem avaliado. Neste contexto, é altamente recomendável que: (*i*) todos os envolvidos tenham suas responsabilidades e obrigações legais pelo cumprimento e gerenciamento de processos de segurança definidos; (*ii*) haja um conjunto consistente de requisitos de segurança a ser implementado em um sistema de informação; (*iii*) execução de auditoria para checar se a segurança do software foi conseguida.

A atividade "Executar métodos de identificação de vulnerabilidade de segurança" obteve o valor de 3,52, evidenciando sua posição entre *importante* e *muito importante*. Portanto, métodos, técnicas, e critérios devem ser executados para identificar e caracterizar as vulnerabilidades de segurança do software em um ambiente definido.

A Figura 5.6 apresenta as três atividades que obtiveram os menores valores, em relação ao seu grau de importância, na avaliação de um processo seguro, segundo os especialistas em segurança da informação.

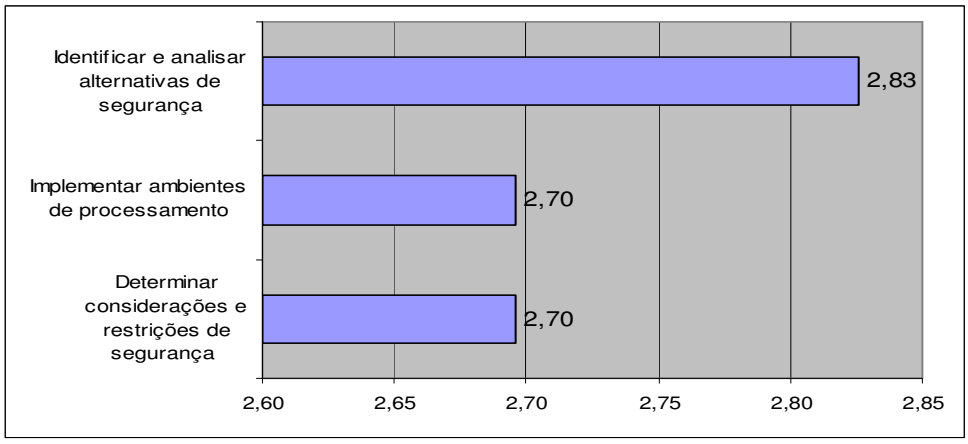

Figura 5.6 As atividades de menor grau de avaliação por especialistas em segurança da informação

As atividades "Determinar considerações e restrições de segurança" e "Implementar ambientes de processamento" foram consideradas de menor grau de importância em um processo seguro de software, obtendo o valor de 2,70, na opinião dos especialistas em segurança. Isto significa que esses especialistas deram importância mediana à questão da identificação de considerações e restrições, que possam influenciar nas tomadas de decisão do processo, bem como na implementação de ambientes de desenvolvimento, ambientes de ensaio e ambientes de produção.

A atividade "Identificar e analisar alternativas de segurança" também teve uma avaliação de importância mediana com valor de 2,83. Assim sendo, é considerada de média importância, com tendência para importante, a identificação e análise de soluções alternativas para problemas relacionados com segurança.

5.4 Aspectos que Influenciam a Utilização de um Processo Seguro

Este trabalho propõe um conjunto de aspectos que poderiam influenciar a utilização de um processo seguro.

A Tabela 5.2 mostra, em ordem decrescente do grau de importância (média aritmética e desvio padrão), esses aspectos que foram analisados pelos especialistas desta pesquisa de campo.

Esses aspectos representam ações estratégicas e organizacionais que podem chegar a promover um processo de apoio para o desenvolvimento de software mais seguro.

O aspecto "Alto comprometimento da gerência com o projeto" foi o melhor avaliado com o valor de 3,69, tendendo para muito influente na utilização de um processo seguro. Isto evidencia a relevância do comprometimento dos gestores de negócio e gestores de tecnologia com o projeto onde será implantada a prática de segurança no desenvolvimento de software.

Tabela 5.2: Aspectos que influenciam a utilização do processo seguro avaliados pelos especialistas

Aspectos que Influenciam a Utilização de um Processo Seguro	Média	Desvio padrão
Alto comprometimento da gerência com o projeto	3,69	0,60
Apoio da direção da empresa	3,66	0,61
O processo seguro estar alinhado aos objetivos de negócio da organização	3,55	0,78
O processo seguro estar baseado em expectativas realistas	3,45	0,69
Equipe com experiência em segurança da informação	3,41	0,78
Responsabilidades claramente definidas	3,38	0,82
Gerência com experiência em segurança da informação	3,28	0,96
Treinamento formal da equipe no processo seguro	3,24	0,74
Existência de orientações para uso do processo seguro	3,10	0,82
Disciplina na implantação das atividades	2,93	0,88
Existência de auditorias da aderência ao processo seguro	2,90	0,86
Iniciar a implantação do processo seguro por um conjunto específico de atividades	2,86	0,74
Existência de um grupo de Segurança da Informação na empresa	2,83	0,80
Ambiente físico de trabalho adequado	2,76	0,95
Participação do cliente nas etapas do processo	2,72	1,10
Apoio à utilização do conhecimento de experiências em projetos anteriores	2,72	0,65
Iniciar a implantação do processo seguro de maneira uniforme	2,62	1,01
Existência de apoio automatizado	2,24	0,83
Apoio da direção da empresa	3,66	0,61
O processo seguro estar alinhado aos objetivos de negócio da organização	3,55	0,78
O processo seguro estar baseado em expectativas realistas	3,45	0,69
Equipe com experiência em segurança da informação	3,41	0,78

Responsabilidades claramente definidas	3,38	0,82
Gerência com experiência em segurança da informação	3,28	0,96
Treinamento formal da equipe no processo seguro	3,24	0,74
Existência de orientações para uso do processo seguro	3,10	0,82
Disciplina na implantação das atividades	2,93	0,88
Existência de auditorias da aderência ao processo seguro	2,90	0,86
Iniciar a implantação do processo seguro por um conjunto específico de atividades	2,86	0,74
Existência de um grupo de Segurança da Informação na empresa	2,83	0,80
Ambiente físico de trabalho adequado	2,76	0,95
Participação do cliente nas etapas do processo	2,72	1,10
Apoio à utilização do conhecimento de experiências em projetos anteriores	2,72	0,65
Iniciar a implantação do processo seguro de maneira uniforme	2,62	1,01
Existência de apoio automatizado	2,24	0,83

O aspecto "Apoio da direção da empresa" foi o segundo melhor avaliado com o valor de 3,66, convergindo para o patamar de *muito influente* na utilização de um processo seguro. Isto reforça a relevância do comprometimento dos gestores, nesse caso a alta direção, com o projeto onde será implantada a prática de segurança no desenvolvimento de software.

O aspecto "O processo seguro estar alinhado aos objetivos de negócio da organização" foi o terceiro melhor avaliado com o valor de 3,55, próximo do valor de *muito influente* estabelecido para a utilização de um processo seguro. Isto ratifica a necessidade de ter os objetivos da segurança de software, pela utilização de um processo seguro, alinhados com os objetivos de negócio da organização. Esse aspecto oferece maiores garantias de que o processo seguro de desenvolvimento satisfará as estratégias de negócio.

O aspecto "O processo seguro estar baseado em expectativas realistas" foi o quarto melhor avaliado com o valor de 3,45, posicionando-se entre a escala de valor influente e valor muito influente para a utilização de um processo seguro. Isto ressalta a necessidade de definir expectativas realistas baseadas em um escopo adequado para se concluir eficazmente o projeto e para obter sucesso na implantação de um processo seguro.

O aspecto "Existência de apoio automatizado" foi considerado de média influência na utilização de um processo seguro e o que obteve menor avaliação, obtendo o valor de 2,24. Isto significa que os especialistas deram pouca influência à questão de possuir ferramentas automatizadas para apoiar a implantação de um processo seguro.

O aspecto "Iniciar a implantação do processo seguro de maneira uniforme" também recebeu uma avaliação de influência mediana, contudo mais próxima de influente, com valor de 2,62. Assim sendo, uma alternativa para implantar inicialmente um processo seguro, e de forma não uniforme, seria especializá-lo para a organização que o utilizar, segundo suas

necessidades e peculiaridades. Ou seja, pode-se utilizar apenas um subconjunto de suas atividades.

5.5 Estruturação do Processo Seguro

A partir dos resultados da pesquisa de campo realizada, o processo seguro proposto foi finalmente estruturado. Houve redução de 40 atividades propostas inicialmente para 37 atividades, pois algumas atividades foram refinadas e integradas em seu conteúdo ou sua abrangência, como apresentado a seguir:

1. "Definir objetivos de planejamento de segurança e identificar seus mecanismos" foi renomeada para "Desenvolver plano de segurança".
2. "Atribuir responsabilidades de segurança no projeto" foi integrada à atividade "Desenvolver plano de segurança".
3. "Implementar ambientes de processamento" foi renomeada para "Planejar ambientes de processamento".
4. "Executar métodos de identificação de vulnerabilidade de segurança" foi renomeada para "Identificar vulnerabilidades de segurança".
5. "Identificar as ameaças de segurança aos ativos críticos" foi renomeada para "Identificar as ameaças de segurança".
6. "Classificar as ameaças de segurança aos ativos" foi renomeada para "Classificar as ameaças de segurança".
7. "Priorizar processos críticos influenciados pelo sistema" foi renomeada para "Tratar as atividades críticas para segurança".
8. "Revisar ativos do sistema que se referem à segurança" foi renomeada para "Revisar artefatos do software que impactam na segurança".
9. "Capturar uma visão de alto nível orientada à segurança da operação do sistema" foi renomeada para "Capturar uma visão de alto nível orientada à segurança do software".
10. "Entender e revisar necessidades de informação de segurança" foi renomeada para "Entender necessidades de informação de segurança".
11. "Determinar considerações e restrições de segurança" foi renomeada para "Identificar restrições e ponderações de segurança".

12. “Identificar e analisar alternativas de segurança” foi renomeada para “Fornecer alternativas de segurança”.
13. “Fornecer orientação de segurança” foi removida do processo seguro.
14. “Identificar e revisar requisitos de garantia de segurança” foi renomeada para “Fornecer requisitos de apoio ao processo”.
15. “Gerenciar e controlar serviços e componentes operacionais de segurança” foi renomeada para “Gerenciar serviços e componentes de segurança”.
16. “Gerenciar percepção, treinamento e programa de educação de segurança” foi renomeada para “Gerenciar treinamento e programas de educação de segurança”.
17. “Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, no ambiente, e em suas características” foi renomeada para “Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, e no ambiente”.
18. “Reavaliar mudanças nas ameaças, vulnerabilidades, impactos, riscos e no ambiente” foi integrada à atividade “Realizar auditorias de segurança”.
19. “Definir estratégia de manutenção da garantia de segurança” foi renomeada para “Definir estratégia de garantia de segurança”.
20. “Conduzir análise de impacto de segurança das mudanças” foi renomeada para “Conduzir análise de impacto das mudanças na segurança”.
21. “Controlar as evidências da manutenção da garantia de segurança” foi renomeada para “Controlar as evidências da garantia de segurança”.
22. “Revisar o comportamento de segurança do sistema para identificar mudanças necessárias” foi renomeada para “Revisar o comportamento de segurança do software para identificar mudanças necessárias”.

As seguintes atividades foram mantidas: “Planejar o gerenciamento de incidentes de segurança”, “Analisar as vulnerabilidades de segurança identificadas”, “Desenvolver estratégias de redução das ameaças de segurança”, “Identificar e descrever impactos de segurança”, “Identificar exposição de segurança”, “Avaliar risco de exposição de segurança”, “Priorizar riscos de segurança”, “Compreender as necessidades de segurança do cliente”, “Definir requisitos de segurança”, “Obter acordo sobre requisitos de segurança”, “Definir a abordagem de verificação e validação de segurança”, “Realizar verificação de segurança”,

"Realizar validação de segurança", "Revisar e comunicar resultados de verificação e validação de segurança", "Gerenciar a implementação de controles de segurança", "Analisar registro de evento com impacto na segurança", e "Identificar e preparar a resposta dos incidentes de segurança relevantes".

5.6 Conclusão

Este capítulo apresentou e analisou os resultados de uma pesquisa de campo realizada com especialistas em processo de desenvolvimento de software e especialistas em segurança da informação, considerando um conjunto de atividades para o Processo de Apoio à Segurança de Software.

Destacam-se dois pontos importantes percebidos com o resultado da pesquisa e que forneceram maior garantia na adequada condução da pesquisa:

- As diversas coincidências no resultado das avaliações dos especialistas em processo de software e dos especialistas em segurança da informação.
- Apesar de haver atividades menos e mais importantes, as diferenças identificadas em todos os resultados das avaliações não são muito significativas entre a atividade pior pontuada e a atividade melhor pontuada. A maior diferença de valores, de 2,74 a 3,72, obtida na avaliação conjunta de todos os especialistas, conforme Tabela 5.1, demonstra que mesmo as "piores" atividades ainda têm certa importância para o processo. Devido a essa constatação, foram mantidas as 37 atividades no processo final.

No capítulo seguinte, com base nessa pesquisa, será apresentado o processo proposto de apoio à segurança dos produtos de software.

Capítulo **6**

Processo de Apoio à Segurança de Software

Este capítulo propõe o Processo de Apoio à Segurança de Software, baseado em abordagens relevantes em segurança da informação.

O Processo de Apoio à Segurança de Software (PASS) proposto, ou simplesmente Processo de Apoio, trata da segurança da informação, a partir de um conjunto de atividades, relacionadas no capítulo 5, a serem executadas ao longo do ciclo de vida, visando melhorar a abordagem da segurança da informação em projetos de software.

6.1. Objetivos do Processo de Apoio

O processo de apoio está estruturado de modo a atender aos seguintes objetivos:

- Prover mais visibilidade para a segurança da informação em um processo de desenvolvimento de *software*.
- Tratar sistematicamente vulnerabilidades, ameaças, impactos e riscos relacionados à segurança do produto de software.
- Possibilitar que ações de segurança estejam alinhadas aos objetivos estratégicos especificados para a organização.

6.2. Papéis e Responsabilidade

Para o processo de apoio são propostos dois papéis e suas respectivas responsabilidades:

- *Engenheiro de Segurança*: especializa e instancia o processo de apoio para os objetivos do projeto, alinhado com metas e planos da organização, e monitora se esses projetos satisfazem os objetivos de segurança.
- *Auditor de Segurança*: avalia a aderência do processo de apoio nos projetos de *software*, em termos de resultados das atividades, dos artefatos produzidos, verificando a conformidade das ações dos projetos ao processo estabelecido.

6.3. Fluxo do Processo de Apoio

O fluxo do processo de apoio é apresentado na Figura 6.1, sendo constituído dos seguintes subprocessos: (i) Planejar Segurança; (ii) Avaliar Vulnerabilidade de Segurança; (iii) Modelar Ameaça de Segurança; (iv) Avaliar Impacto de Segurança; (v) Avaliar Risco de Segurança; (vi) Especificar Necessidades de Segurança; (vii) Fornecer Informação de Segurança; (viii) Verificar e Validar Segurança; (ix) Gerenciar Segurança; (x) Garantir Segurança; e (xi) Monitorar Comportamento de Segurança.

A Figura 6.1 representa também os subprocessos através de cores que, conforme a legenda, são propostas genéricas que tentam associar os subprocessos com algumas disciplinas do RUP (2003). Essa representação é uma sugestão de trabalho futuro cuja idéia macro é identificar onde o PASS poderia contribuir com a inclusão de novas atividades nas disciplinas ou fases do RUP.

O processo proposto adequa-se ao ciclo de vida iterativo incremental, e seus subprocessos são compostos por um conjunto de atividades. Cada atividade é descrita em termos de propósito, tarefas, artefatos de entrada, artefatos de saída, e atores envolvidos. Esta estrutura está baseada no RUP (2003). Para a utilização de outros ciclos de vida, isto precisaria ser validado.

Os subprocessos de segurança elencados representam uma proposta não exaustiva. Neste contexto, é recomendável especializar o processo proposto para a organização que o utilizar, segundo suas necessidades e peculiaridades. Ou seja, é possível utilizar apenas um subconjunto dos subprocessos e ou atividades propostos, adaptá-los e ou incluir novas atividades específicas.

A partir do processo de apoio especializado para a organização, poderá haver então instanciação desse para a execução dos projetos de software. Na instanciação do processo de apoio para os projetos, nem todos os artefatos podem ser requeridos ou gerados em cada atividade. Isto dependerá do projeto em questão. A seguir, serão detalhados os subprocessos do processo de apoio.

6.3.1. Subprocesso: Planejar Segurança

O propósito do subprocesso Planejar Segurança (Figura 6.2) é garantir que as informações necessárias para o planejamento da segurança para o projeto sejam estabelecidas e registradas.

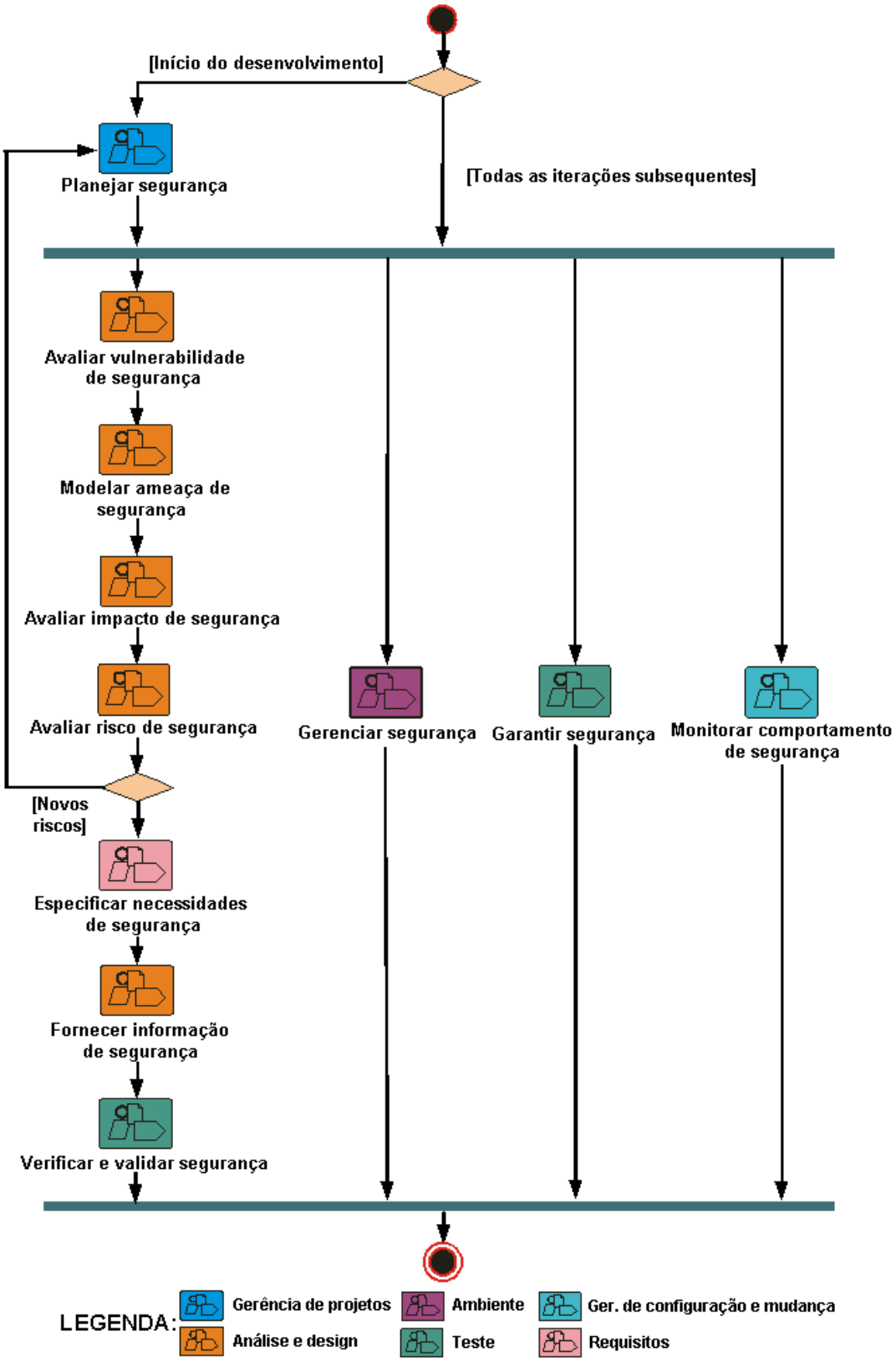

Figura 6.1: Processo de Apoio à Segurança de Software

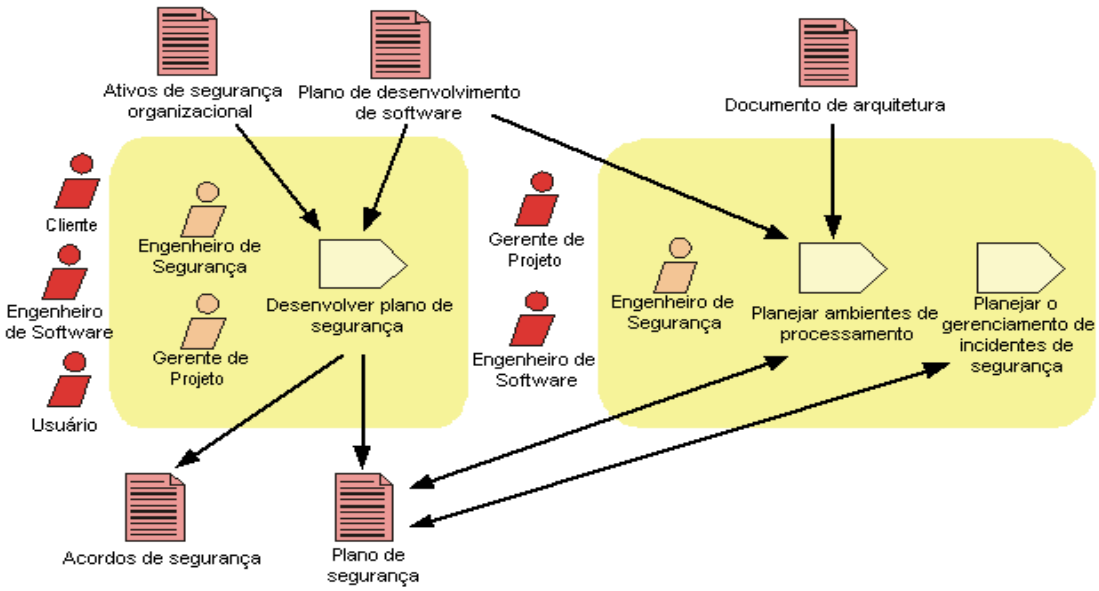

Figura 6.2: Subprocesso: Planejar Segurança

Devem ser elaborados (ou refinados) os objetivos e o plano de segurança da informação para a organização, e ser constituída a equipe de segurança (quando da instanciação do processo de apoio).

O engenheiro de segurança é responsável por definir junto à equipe do projeto qual a visão de segurança a ser estabelecida para o projeto. Atua também como consultor de segurança de software para essa equipe.

6.3.1.1 Artefatos

A Tabela 6.1 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.1: Artefatos do subprocesso "Planejar Segurança"

Artefato	Descrição
Documento de arquitetura	Documenta uma visão geral de arquitetura abrangente do sistema, usando diversas visões de arquitetura para descrever diferentes aspectos do sistema.
Ativos de segurança organizacional	Documenta questões legais, autorizações, e outros documentos de segurança da informação. Fazem parte dos ativos organizacionais. Pode ser apresentado em uma seção do Plano de segurança.
Plano de desenvolvimento de software	Documenta as informações necessárias ao gerenciamento do projeto.
Acordos de segurança	Documenta os acordos de segurança que influenciam a realização do projeto. Ex.: acordo de confidencialidade, acordo de troca de informação, acordo de nível de serviço, etc.
Plano de segurança	Documenta os vários aspectos do planejamento de segurança para o projeto de software.

Este subprocesso é composto pelas seguintes atividades: (i) Desenvolver Plano de segurança; (ii) Planejar ambientes de processamento; e (iii) Planejar o gerenciamento de incidentes de segurança.

6.3.1.2 Atividade: Desenvolver Plano de Segurança

O Quadro 6.1 apresenta o detalhamento desta atividade.

Atividade: Desenvolver Plano de Segurança	
Propósito: definir o plano de segurança, identificando mecanismos de coordenação do projeto.	
Tarefa: - Desenvolver plano de segurança	
Artefatos deEntrada: - Plano de desenvolvimento de software - Ativos de segurança organizacional	Artefatos Resultantes: - Acordos de segurança - Plano de segurança
Ator: Engenheiro de Segurança, Engenheiro de Software, Gerente de Projeto, Cliente, Usuário	

Quadro 6.1: Atividade: Desenvolver plano de segurança

Esta atividade possui a tarefa "Desenvolver plano de segurança".

Os objetivos de segurança da organização devem estar formalmente documentados nos "Ativos de segurança organizacional", e servem de base, juntamente com o Plano de desenvolvimento de software, para a definição de objetivos e estratégias de segurança para o projeto.

Deve ser estruturada a equipe de segurança do projeto, incluindo o engenheiro de segurança e o auditor de segurança, podendo ser incluídos também outros profissionais que possam vir a auxiliar na implantação do processo de apoio. Relacionamentos e compromissos com a equipe do processo de desenvolvimento devem ser acordados, assegurando que as linhas de comunicação estejam bem estabelecidas.

6.3.1.3 Atividade: Planejar Ambientes de Processamento

O Quadro 6.2 apresenta o detalhamento desta atividade.

Atividade: Planejar Ambientes de Processamento.	
Propósito: identificar e analisar os ambientes de desenvolvimento, ensaio e produção.	
Tarefa: - Planejar ambientes de processamento	
Artefatos de Entrada: - Plano de desenvolvimento de software - Documento de arquitetura - Plano de segurança	Artefatos Resultantes: - Plano de segurança (ambientes de processamento)
Ator: Engenheiro de Segurança, Engenheiro de Software, Gerente de Projeto	

Quadro 6.2: Atividade: Planejar ambientes de processamento

Esta atividade possui a tarefa “Planejar ambientes de processamento”.

O ambiente de desenvolvimento, o ambiente de ensaio e o ambiente de produção são identificados e analisados, avaliando-se o nível de separação necessário entre eles para prevenir problemas operacionais.

6.3.1.4 Atividade: Planejar o Gerenciamento de Incidentes de Segurança

O Quadro 6.3 apresenta o detalhamento desta atividade.

Atividade: Planejar o Gerenciamento de Incidentes de Segurança	
Propósito: o projeto é analisado visando o planejamento para gerenciar os incidentes de segurança.	
Tarefa: - Planejar o gerenciamento de incidentes de segurança	
Artefatos de Entrada: - Plano de segurança	Artefatos Resultantes: - Plano de segurança (gerenciamento de incidentes de segurança).
Ator: Engenheiro de Segurança, Engenheiro de Software, Gerente de Projeto	

Quadro 6.3: Atividade: Planejar o gerenciamento de incidentes de segurança

Esta atividade possui a tarefa “Planejar o gerenciamento de incidentes de segurança”.

O plano de gerenciamento de incidentes é definido para o projeto, considerando os ambientes identificados e analisados, pra garantir uma resposta rápida, efetiva, e ordenada aos incidentes de segurança quando ocorrerem.

6.3.2. Subprocesso: Avaliar Vulnerabilidade de Segurança

O propósito do subprocesso Avaliar Vulnerabilidade de Segurança (Figura 6.3) é identificar e caracterizar (na iteração) vulnerabilidades de segurança do software relacionadas com um ambiente definido, normalmente no ambiente onde o sistema será utilizado.

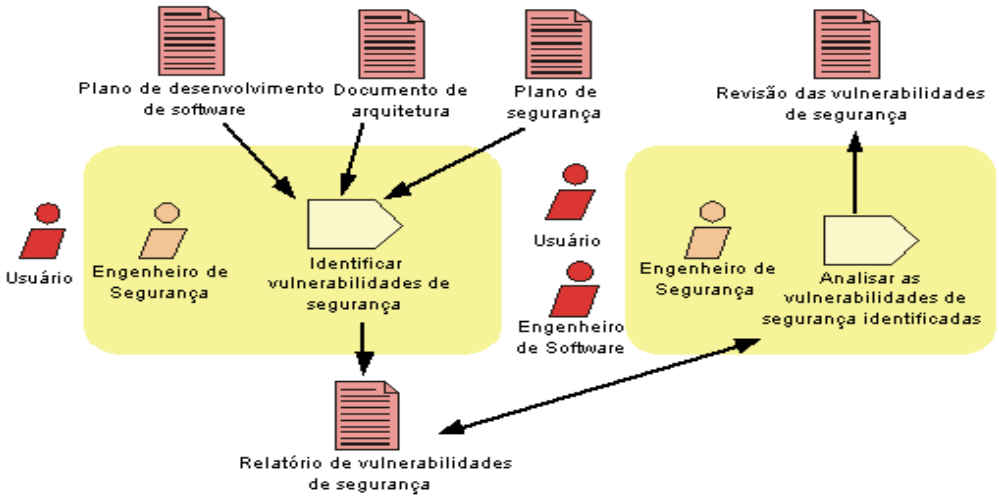

Figura 6.3 - Subprocesso: Avaliar Vulnerabilidade de Segurança

Neste subprocesso, são definidas as vulnerabilidades de segurança do software, fornecendo-se uma avaliação geral dessas vulnerabilidades na iteração em questão.

O Engenheiro de Segurança é responsável por conduzir os métodos de identificação de vulnerabilidades de segurança e analisá-las, a partir de discussões conjuntas com os principais usuários (membros dos níveis estratégico (executivos), tático (gerência), e operacional (técnico)) e com o apoio do engenheiro de software.

6.3.2.1 Artefatos

A Tabela 6.2 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.2: Artefatos do subprocesso "Avaliar Vulnerabilidade de Segurança"

Artefato	Descrição
Documento de arquitetura	Ver Tabela 6.1.
Plano de desenvolvimento de software	Ver Tabela 6.1.
Plano de segurança	Ver Tabela 6.1.
Relatório de vulnerabilidades de segurança	Documenta as vulnerabilidades de segurança identificadas, suas propriedades e a relação de perigos (quando aplicável).
Revisão das vulnerabilidades de segurança	Documenta a revisão gerada pela análise das vulnerabilidades identificadas, incluindo as ações de curto prazo para eliminar as vulnerabilidades mais críticas. Pode vir a ser uma seção do relatório de vulnerabilidades de segurança.

Este subprocesso é composto pelas seguintes atividades: (i) Identificar vulnerabilidades de segurança; e (ii) Analisar as vulnerabilidades de segurança identificadas.

6.3.2.2 Atividade: Identificar Vulnerabilidades de Segurança

O Quadro 6.4 apresenta o detalhamento desta atividade.

Atividade: Identificar Vulnerabilidades de Segurança	
Propósito: executar métodos, técnicas, e critérios pelos quais as vulnerabilidades de segurança do software em um ambiente definido são identificadas e caracterizadas.	
Tarefas: - Implementar análise de perigo - Analisar a força das funções de segurança	
Artefatos de Entrada: - Plano de desenvolvimento de software - Documento de arquitetura - Plano de segurança	Artefatos Resultantes: - Relatório de vulnerabilidades de segurança
Ator: Engenheiro de Segurança, Usuário	

Quadro 6.4: Atividade: Identificar vulnerabilidades de segurança

Essa atividade consiste na execução de métodos para identificar e caracterizar as vulnerabilidades de segurança de software. Isto pode incluir um esquema para categorizar e priorizar as vulnerabilidades baseadas em probabilidade, funções operacionais, regulamentações ou outros critérios, quando fornecido. Esta atividade possui as duas tarefas abaixo.

Implementar Análise de Perigo

A análise de perigo envolve um conjunto de técnicas para expor estados de perigo do processo de negócio do software, como pelo uso inapropriado. Busca causas prováveis de problemas de tal forma que se possa aplicar uma técnica apropriada para prevenir o problema ou amenizar suas conseqüências. Pode envolver o exame de caminhos de acesso ou interfaces de ativos críticos, análise de canais ocultos, o desenvolvimento de listas de perigo, bem como procedimentos para explorar cenários hipotéticos, para considerar perigos não óbvios. São levantados os impactos potenciais para a organização, baseados nesses cenários de perigo.

Um canal oculto (*covert channel*) pode tornar-se um fluxo ilícito de informação. A análise desses canais procura descobrir sua existência e a sua capacidade de exploração.

Investiga se o software pode ser configurado ou utilizado de maneira insegura, mesmo que seus administradores e usuários acreditem que tudo esteja seguro. Procura: (i) minimizar a probabilidade de configurar ou instalar o software de maneira que seja inseguro, sem que o usuário e o administrador sejam capazes de detectar o problema; e (ii) minimizar o risco de que erros operacionais, humanos ou não, causem desativação, desabilitação, ou falha em ativar funções de segurança, resultando em um estado inseguro não detectável.

Analisar a Força das Funções de Segurança

Mesmo que as funções de segurança do software não possam ser contornadas, desativadas ou corrompidas, pode ainda ser possível afetá-las porque há ainda vulnerabilidades escondidas nos mecanismos de segurança. Para essas funções, a qualificação de seus comportamentos seguros pode ser obtida utilizando-se resultados quantitativos ou estatísticos do comportamento de segurança desses mecanismos e o esforço necessário para contorná-los.

A análise deve indicar se o mecanismo de segurança atinge ou ultrapassa sua força. Por exemplo: a análise da força de uma função relacionada ao mecanismo de senhas pode demonstrar que essa função alcança a segurança pretendida ao mostrar que o tamanho da senha e suas regras de formação garantam senhas de difícil identificação e quebra.

O Engenheiro de Segurança verifica as evidências apresentadas e sua coerência, isto é, verifica se as afirmações de força exigidas estão corretas.

6.3.2.3 Atividade: Analisar as Vulnerabilidades de Segurança Identificadas

O Quadro 6.5 apresenta o detalhamento desta atividade.

<table>
<tr><th colspan="2">Atividade: Analisar as Vulnerabilidades de Segurança Identificadas</th></tr>
<tr><td colspan="2">Propósito: as vulnerabilidades de segurança do software identificadas são analisadas.</td></tr>
<tr><td colspan="2">Tarefas:
- Coletar dados de vulnerabilidades de segurança
- Revisar vulnerabilidades de segurança
- Planejar ações para as vulnerabilidades de segurança</td></tr>
<tr><td>Artefatos de Entrada:
- Relatório de vulnerabilidades de segurança</td><td>Artefatos Resultantes:
- Relatório de vulnerabilidades de segurança (análise das vulnerabilidades identificadas)
- Revisão das vulnerabilidades de segurança</td></tr>
<tr><td colspan="2">Ator: Engenheiro de Segurança, Engenheiro de Software, Usuário</td></tr>
</table>

Quadro 6.5: Atividade: Analisar as vulnerabilidades de segurança identificadas

A atividade consiste de uma avaliação que determina se vulnerabilidades de segurança identificadas durante a atividade anterior podem permitir que usuários violem as funcionalidades de segurança, como conseguir acesso não autorizado aos recursos, interferir ou alterar as funcionalidades de segurança, ou interferir com as autorizações de outros usuários.

Vulnerabilidades podem ser encontradas em partes do software relacionadas com segurança ou não. Em muitos casos, mecanismos não relacionados com segurança, mas que apóiem funções de segurança ou trabalhem com mecanismos de segurança possuem vulnerabilidades exploráveis. Os cenários de perigo desenvolvidos na atividade anterior deveriam ser seguidos para que as vulnerabilidades sejam validadas. As vulnerabilidades descobertas do software deveriam ser registradas. Esta atividade possui as três tarefas abaixo.

Coletar dados de vulnerabilidades de segurança

Coleta de dados relacionados com as propriedades das vulnerabilidades, para identificar, entre outras informações, a porção mais vulnerável (*weakest link*) do software ou artefato.

Revisar vulnerabilidades de segurança

Faz a revisão das vulnerabilidades de segurança do software, agregando vulnerabilidades que resultem de vulnerabilidades específicas e de suas combinações. Revisa

que vulnerabilidades ou combinação de vulnerabilidades resultam em problemas para o software.

Procura identificar características adicionais de cada vulnerabilidade, tais como a probabilidade de explorá-la e a chance para que essa exploração tenha sucesso. Recomendações para tratar as vulnerabilidades produzidas podem também ser incluídas nos resultados.

Planejar ações para as vulnerabilidades de segurança

Elaborar um conjunto de ações que possam ser realizadas para atacar as vulnerabilidades de segurança, especialmente as mais críticas ou de curto prazo.

6.3.3. Subprocesso: Modelar Ameaça de Segurança

O propósito do subprocesso Modelar Ameaça de Segurança (Figura 6.4) é identificar e caracterizar ameaças de segurança com suas propriedades e características, a partir das vulnerabilidades resultantes do subprocesso anterior.

Uma ameaça pode ser definida como um evento que pode comprometer o funcionamento habitual do software, impactando seus objetivos em várias escalas. A modelagem de ameaças pode ser usada para identificar ameaças e guiar decisões de projeto (incluindo artefatos críticos produzidos ao longo do ciclo de vida), evidenciando as ameaças com maior potencial de causar o máximo dano à aplicação.

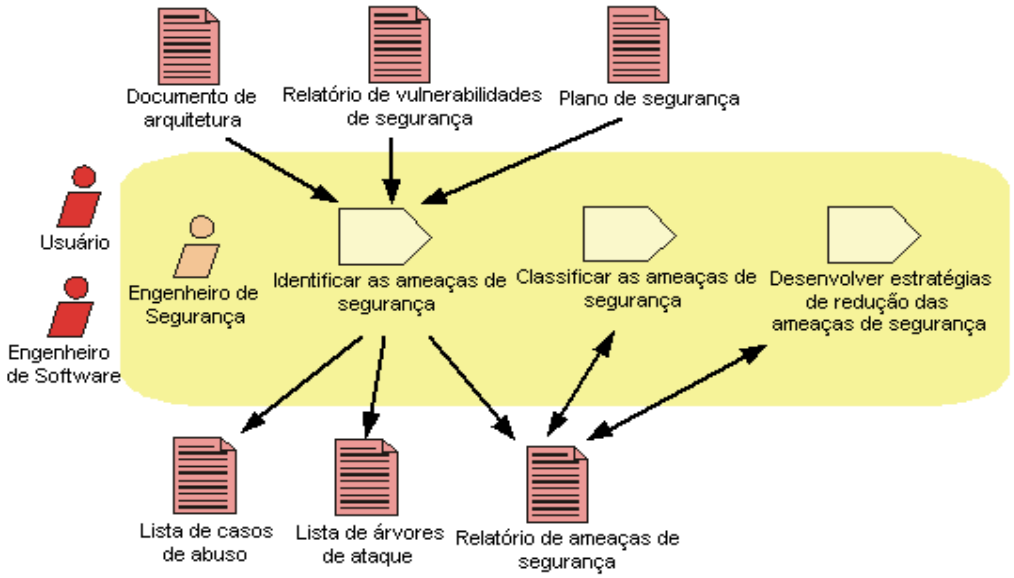

Figura 6.4 - Subprocesso: Modelar Ameaça de Segurança

O Engenheiro de Segurança é responsável por conduzir a identificação das ameaças de segurança, na iteração, implementando casos de abuso e árvores de ataque, classificá-las, e desenvolver estratégias de redução das ameaças, a partir de discussões conjuntas com os principais usuários (membros dos níveis estratégico (executivos), tático (gerência), e operacional (técnico)) e apoio do engenheiro de software.

6.3.3.1 Artefatos

A Tabela 6.3 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.3: Artefatos do subprocesso "Modelar Ameaça de Segurança"

Artefato	Descrição
Plano de segurança	Ver Tabela 6.1.
Documento de arquitetura	Ver Tabela 6.1.
Relatório de vulnerabilidades de segurança	Ver Tabela 6.2.
Relatório de ameaças de segurança	Documenta as ameaças identificadas, sua classificação, e suas estratégias de mitigação.
Lista de casos de abuso	Documenta a relação de casos de abuso desenvolvidos na iteração. Pode vir a ser uma seção do Relatório de ameaças de segurança.
Lista de árvores de ataque	Documenta a relação de árvores de ataque desenvolvidas na iteração. Pode vir a ser uma seção do Relatório de ameaças de segurança.

Este subprocesso é composto pelas seguintes atividades: (i) Identificar as ameaças de segurança; (ii) Classificar as ameaças de segurança; e (iii) Desenvolver estratégias de redução das ameaças de segurança.

6.3.3.2 Atividade: Identificar as Ameaças de Segurança

O Quadro 6.6 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Identificar as Ameaças de Segurança</td></tr>
<tr><td colspan="2">Propósito: identificar as ameaças de segurança de cada artefato crítico ao longo do ciclo de vida do software.</td></tr>
<tr><td colspan="2">Tarefas:
- Implementar casos de abuso
- Implementar árvores de ataque</td></tr>
<tr><td>Artefatos de Entrada:
- Documento de arquitetura
- Relatório de vulnerabilidades de segurança
- Plano de segurança</td><td>Artefatos Resultantes:
- Lista de casos de abuso
- Lista de árvores de ataque
- Relatório de ameaças de segurança</td></tr>
<tr><td colspan="2">Ator: Engenheiro de Segurança, Usuário, Engenheiro de Software</td></tr>
</table>

Quadro 6.6: Atividade: Identificar as ameaças de segurança

Em relação às ameaças causadas por pessoas, há basicamente dois tipos: aquelas que surgem de fontes acidentais, e aquelas que resultam de um ato deliberado. Contudo, deve-se focar em um grupo de ameaças amplo e bem definido, para garantir que as ameaças causadas por pessoas sejam tratadas pelo software. O levantamento dos agentes que exploram as ameaças, dos mecanismos utilizados na exploração e dos artefatos (ativos) físicos e lógicos é crítico para a identificação das ameaças do software. Esta atividade possui as duas tarefas abaixo.

Implementar casos de abuso

Os casos de abuso cobrem práticas que corrompem a segurança do software, descrevendo o comportamento do software sob ataque. Construí-los exige cobertura explicita do que deve ser protegido, de quem, e por quanto tempo. Segundo McGraw (2006), a forma mais simples de criar casos de abuso é por meio de "*brainstorming*". A Figura 6.5. apresenta um exemplo simples de um diagrama de caso de abuso de um software de compras pela Internet.

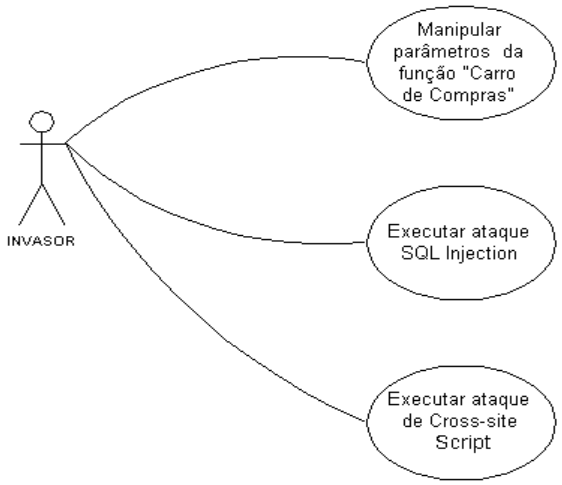

Figura 6.5 – Diagrama de caso de abuso de software de compra pela Internet

Implementar árvores de ataque

As árvores de ataque modelam o processo de decisão para realizar ataques. A raiz representa o objetivo do atacante, como por exemplo, roubar número do cartão de crédito. Os nós representam ações que o atacante realiza, e cada caminho na árvore representa um único ataque para atingir este objetivo (SCHNEIER, 1999). Não existe um padrão único para representar árvores de ataque.

Elas apóiam a análise de riscos, para responder questões a cerca da segurança do software, capturando conhecimento de segurança de forma a reutilizá-lo e para projetar,

implementar e testar proteções contra os ataques. A Figura 6.6. abaixo apresenta um exemplo de uma árvore de ataque.

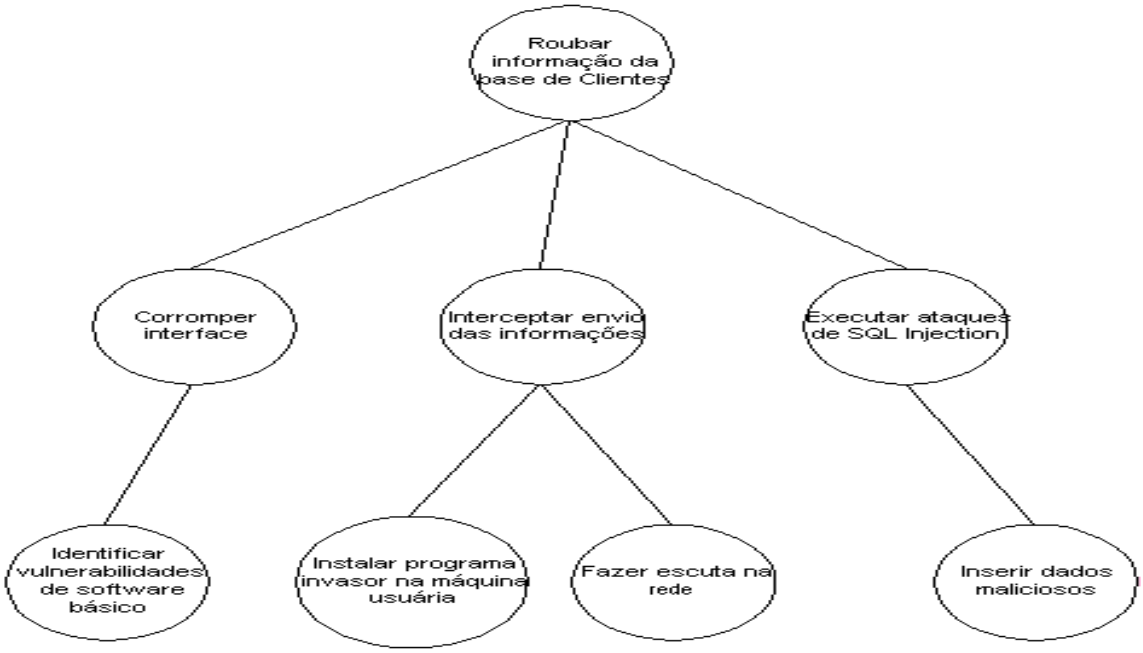

Figura 6.6 – Exemplo de árvore de ataque

6.3.3.3 Atividade: Classificar as Ameaças de Segurança

Baseado nas informações dos agentes da ameaça, na probabilidade das ameaças e em outras informações consideradas relevantes pela equipe de segurança, classificam-se as ameaças. Pode ser utilizado um método para auxiliar a classificação das ameaças, como o STRIDE (*Spoofing identity, Tampering with data, Repudiation, Information disclosure, Denial of service, Elevation of privilege*) (HOWARD, 2002).

O Quadro 6.7 apresenta o detalhamento desta atividade.

Atividade: Classificar as Ameaças de Segurança	
Propósito: as ameaças são classificadas identificando as mais críticas e verossímeis.	
Tarefas: - Avaliar capacidade do agente da ameaça - Avaliar probabilidade da ameaça de segurança	
Artefatos de Entrada: - Relatório de ameaças de segurança	Artefatos Resultantes: - Relatório de ameaças de segurança (ameaças classificadas)
Ator: Engenheiro de Segurança, Engenheiro de Software, Usuário	

Quadro 6.7: Atividade: Classificar as ameaças de segurança

Esta atividade possui as duas tarefas abaixo.

Avaliar capacidade do agente da ameaça

Determinação da habilidade do agente humano (adversário) em potencial e de sua capacidade de executar um ataque de sucesso contra o software. A habilidade trata do conhecimento de ataque do agente (e.g.: eles têm conhecimento?). A capacidade é uma medida da probabilidade que um agente apto pode executar o ataque (e.g.: eles têm os recursos?).

Avaliar probabilidade da ameaça de segurança

Avalia a probabilidade de ocorrência de evento de ameaça, identificando quão provável é a concretização de um evento de ameaça. Muitos fatores necessitam ser considerados para realizar essa avaliação, envolvendo desde a ocorrência de mudança de um evento natural até o ato acidental ou deliberado de um indivíduo.

6.3.3.4 Atividade: Desenvolver Estratégias de Redução das Ameaças de Segurança

O Quadro 6.8 apresenta o detalhamento desta atividade.

Atividade: Desenvolver Estratégias de Redução das Ameaças de Segurança	
Propósito: elaborar um conjunto de estratégias para reduzir a probabilidade da ocorrência de ameaças.	
Tarefa: - Desenvolver estratégias de redução das ameaças de segurança	
Artefatos de Entrada: - Relatório de ameaças de segurança	Artefatos Resultantes: - Relatório de ameaças de segurança (estratégia de redução das ameaças)
Ator: Engenheiro de Segurança, Engenheiro de Software	

Quadro 6.8: Atividade: Desenvolver estratégias de redução das ameaças de segurança

Esta atividade possui a tarefa "Desenvolver estratégias de redução das ameaças de segurança".

As estratégias de redução das ameaças podem envolver, entre outras ações, a aplicação de controles no software, treinamento para desenvolvedores, programas de conscientização, entre outros.

6.3.4. Subprocesso: Avaliar Impacto de Segurança

O propósito do subprocesso Avaliar Impacto de Segurança (Figura 6.7) é identificar e caracterizar impactos que sejam relevantes com respeito ao software e avaliar a possibilidade da ocorrência desses impactos.

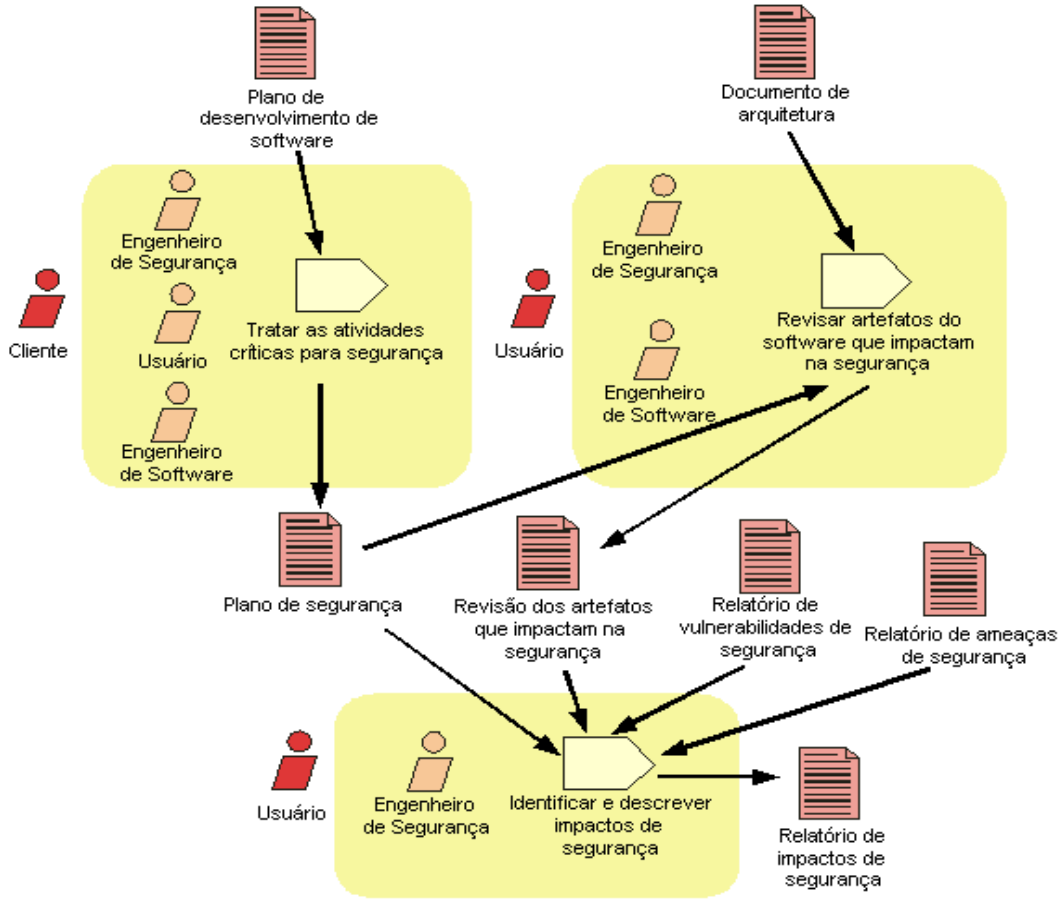

Figura 6.7 - Subprocesso: Avaliar Impacto de Segurança

Os impactos de segurança podem ser tangíveis, como prejuízos financeiros, ou intangíveis, como perda de reputação. Entende-se impacto como sendo a conseqüência de um incidente indesejado, causado tanto deliberada quanto acidentalmente e que afeta o software. As conseqüências podem envolver a destruição ou dano de artefatos do software ou mesmo a perda de sua confidencialidade, integridade, ou disponibilidade.

O Engenheiro de Segurança, Engenheiro de Software, e o Usuário (cliente) são responsáveis pela priorização das atividades críticas influenciadas pelo software. O Engenheiro de Segurança e o Engenheiro de Software tratam da revisão dos artefatos de software que se referem à segurança. A partir das informações anteriores e pela avaliação de cada uma das ameaças e das vulnerabilidades, o Engenheiro de Segurança, sendo apoiado pelo Usuário, identifica os impactos de segurança de incidentes não desejados e descreve informações detalhadas sobre estes, podendo cobrir os custos diretos e em cascata envolvidos por causa dos diferentes riscos de desenvolvimento do produto ou da iteração.

6.3.4.1 Artefatos

A Tabela 6.4 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.4: Artefatos do subprocesso "Avaliar Impacto de Segurança"

Artefato	Descrição
Documento de arquitetura	Ver Tabela 6.1.
Plano de segurança	Ver Tabela 6.1.
Plano de desenvolvimento de software	Ver Tabela 6.1.
Relatório de vulnerabilidades de segurança	Ver Tabela 6.2.
Relatório de ameaças de segurança	Ver Tabela 6.3.
Relatório de impactos de segurança	Documenta e descreve a relação de impactos de segurança identificados.
Revisão dos artefatos que impactam na segurança	Documenta o resultado da revisão dos artefatos do software em desenvolvimento que possuem alto impacto na segurança.

Este subprocesso é composto pelas seguintes atividades: (i) Tratar as atividades críticas para segurança; (ii) Revisar artefatos do software que impactam na segurança; e (iii) Identificar e descrever impactos de segurança.

6.3.4.2 Atividade: Tratar as Atividades Críticas para Segurança

O Quadro 6.9 apresenta o detalhamento desta atividade.

Atividade: Tratar as Atividades Críticas para Segurança	
Propósito: identificar, analisar, e priorizar as atividades críticas (operacionais, de negócio, etc.) do projeto de software.	
Tarefa: - Tratar as atividades críticas para segurança	
Artefatos de Entrada: - Plano de desenvolvimento de software	Artefatos Resultantes: - Plano de segurança
Ator: Engenheiro de Segurança, Engenheiro de Software, Cliente, Usuário	

Quadro 6.9: Atividade: Tratar as atividades críticas para segurança

Esta atividade possui a tarefa "Tratar as atividades críticas para segurança" que identifica, analisa, e prioriza atividades operacionais, de negócio ou relacionadas com a missão da organização cujo comprometimento possa causar grande impacto no software e no negócio da própria organização. A influência das estratégias de negócio deve também ser considerada.

6.3.4.3 Atividade: Revisar Artefatos do Software que Impactam na Segurança

O Quadro 6.10 apresenta o detalhamento desta atividade.

Atividade: Revisar Artefatos do Software que Impactam na Segurança	
Propósito: revisar os artefatos que suportam as capacidades chave ou objetivos de segurança de software e que, por isso, precisam de proteção.	
Tarefa: - Revisar artefatos do software que impactam na segurança	
Artefatos de Entrada: - Documento de arquitetura - Plano de segurança	Artefatos Resultantes: - Revisão dos artefatos que impactam na segurança
Ator: Engenheiro de Segurança, Engenheiro de Software, Usuário	

Quadro 6.10: Atividade: Revisar artefatos do software que impactam na segurança

Esta atividade possui a tarefa "Revisar artefatos do software que impactam na segurança" que revisa os recursos de software e dados necessários para apoiar os objetivos de segurança ou as capacidades chave (funções operacionais, de negócio ou de missão) do software. Cada um desses artefatos deverá ter seu sistema de informação com o respectivo proprietário e a classificação de segurança acordada e documentada. Serão revisados principalmente os artefatos de atividades de maior prioridade.

6.3.4.4 Atividade: Identificar e Descrever Impactos de Segurança

O Quadro 6.11 apresenta o detalhamento desta atividade.

Atividade: Identificar e Descrever Impactos de Segurança	
Propósito: identificar e descrever impactos de segurança de incidentes não desejados.	
Tarefa: - Identificar e descrever impactos de segurança	
Artefatos de Entrada: - Plano de segurança - Revisão dos artefatos que impactam na segurança - Relatório de vulnerabilidades de segurança - Relatório de ameaças de segurança	Artefatos Resultantes: - Relatório de impactos de segurança
Ator: Engenheiro de Segurança, Usuário	

Quadro 6.11: Atividade: Identificar e descrever impactos de segurança

Esta atividade possui a tarefa "Identificar e descrever impactos de segurança".

A partir dos artefatos e capacidades levantados anteriormente, identificam-se os incidentes cujas conseqüências causariam maior dano (maior impacto). A descrição de impacto é uma narrativa de como uma ameaça afeta a missão da organização. A combinação de ameaça, de vulnerabilidade e do impacto resultante na organização define o risco da organização.

6.3.5. Subprocesso: Avaliar Risco de Segurança

O propósito do subprocesso Avaliar Risco de Segurança (Figura 6.8) é, de posse das informações de vulnerabilidade, ameaça e impacto de segurança na iteração, avaliar os riscos inerentes do software em desenvolvimento pela identificação da exposição de segurança, o risco dessa exposição, e a priorização desses riscos.

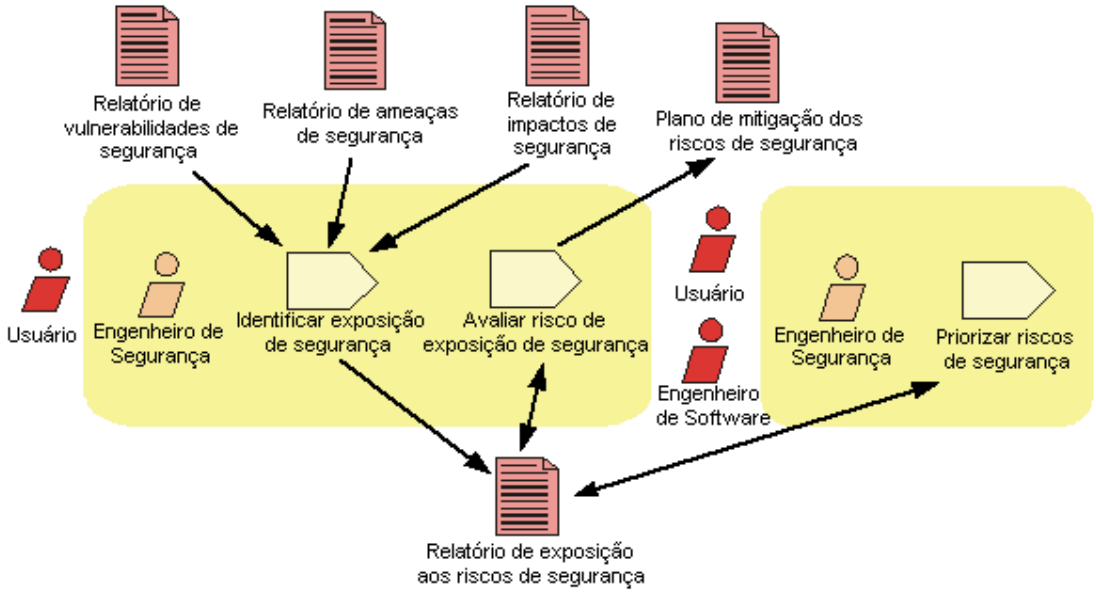

Figura 6.8 - Subprocesso: Avaliar Risco de Segurança

Esse subprocesso foca na avaliação de riscos de segurança relacionados com o software em um ambiente definido, baseados no entendimento estabelecido de como os artefatos são vulneráveis às ameaças levantadas. Isto envolve identificar e avaliar a possibilidade da ocorrência das exposições. Caso existam indícios da existência de novos riscos de segurança, isto deve ser tratado inicialmente no subprocesso "Planejar segurança" com o fito de averiguar se os novos riscos poderiam afetar todo o plano. Em seguida, os outros subprocessos seriam executados para identificar se existem informações de vulnerabilidades, ameaças, e impactos de segurança relacionadas com o novo risco potencial.

O Engenheiro de Segurança, com apoio do Usuário, é responsável pela identificação da exposição de segurança, pela avaliação do risco da exposição, e pela priorização desses riscos. O Engenheiro de Software pode também auxiliar a definir a priorização dos riscos de segurança.

6.3.5.1 Artefatos

A Tabela 6.5 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.5: Artefatos do subprocesso "Avaliar Risco de Segurança"

Artefato	Descrição
Relatório de vulnerabilidades de segurança	Ver Tabela 6.2.
Relatório de ameaças de segurança	Ver Tabela 6.3.
Relatório de impactos de segurança	Ver Tabela 6.4.
Relatório de exposição aos riscos de segurança	Documenta a exposição (ameaça, vulnerabilidade, impacto) ao risco de segurança, a incerteza inerente ao risco, e sua priorização.
Plano de mitigação dos riscos de segurança	Documenta o plano para reduzir o impacto dos riscos sobre os artefatos (ativos de informação do software).

Este subprocesso é composto pelas seguintes atividades: (i) Identificar exposição de segurança; (ii) Avaliar risco de exposição de segurança; e (iii) Priorizar riscos de segurança.

6.3.5.2 Atividade: Identificar Exposição de Segurança

O Quadro 6.12 apresenta o detalhamento desta atividade.

Atividade: Identificar Exposição de Segurança	
Propósito: identificar a tripla ameaça/vulnerabilidade/impacto (exposição) de segurança do software.	
Tarefa: - Identificar exposição de segurança	
Artefatos de Entrada: - Relatório de vulnerabilidades de segurança - Relatório de ameaças de segurança - Relatório de impactos de segurança	Artefatos Resultantes: - Relatório de exposição aos riscos de segurança
Ator: Engenheiro de Segurança, Usuário	

Quadro 6.12: Atividade: Identificar exposição de segurança

Esta atividade possui a tarefa "Identificar exposição de segurança" cujo propósito é reconhecer que ameaças e vulnerabilidades de segurança são relevantes para o software, identificando seus impactos de ocorrência.

6.3.5.3 Atividade: Avaliar Risco de Exposição de Segurança

O Quadro 6.13 apresenta o detalhamento desta atividade.

Atividade: Avaliar Risco de Exposição de Segurança
Propósito: avaliar o risco associado a cada exposição de segurança, através da probabilidade da ocorrência de cada exposição de segurança.
Tarefas: - Avaliar nível geral de incerteza

- Criar planos de mitigação de riscos de segurança	
Artefatos de Entrada: - Relatório de exposição aos riscos de segurança	Artefatos Resultantes: - Relatório de exposição aos riscos de segurança (avaliação do risco de exposição) - Plano de mitigação dos riscos de segurança
Ator: Engenheiro de Segurança, Usuário	

Quadro 6.13: Atividade: Avaliar risco de exposição de segurança

Esta atividade possui as duas tarefas abaixo.

Avaliar nível geral de incerteza

Avalia o nível geral de incerteza associado com o risco de exposição de segurança. Cada risco deve estar associado a um nível de incerteza. O nível geral de incerteza de risco é a "soma" das incertezas que foram identificadas para as ameaças, vulnerabilidades, e impactos de segurança e suas características (seção 6.3.2.3, seção 6.3.3.3 (tarefa "Avaliar a probabilidade da ameaça de segurança"), e seção 6.3.4.4).

Criar plano de mitigação de riscos de segurança

A equipe de segurança cria plano de mitigação de riscos e propõe controles no sistema de proteção para os artefatos críticos. Um plano de mitigação define as atividades ou controles requeridos a serem implementados para mitigar riscos/ameaças de artefatos críticos. A mitigação pode tratar a ameaça, vulnerabilidade, impacto, ou mesmo o risco, sendo conseguida através da implantação de controles de segurança.

6.3.5.4 Atividade: Priorizar Riscos de Segurança

O Quadro 6.14 apresenta o detalhamento desta atividade.

Atividade: Priorizar Riscos de segurança	
Propósito: ordenar riscos de segurança por prioridade.	
Tarefa: - Priorizar riscos de segurança	
Artefatos de Entrada: - Relatório de exposição aos riscos de segurança	Artefatos Resultantes: - Relatório de exposição aos riscos de segurança (riscos priorizados)
Ator: Engenheiro de Segurança, Engenheiro de Software, Usuário	

Quadro 6.14: Atividade: Priorizar riscos de segurança

Esta atividade possui a tarefa "Priorizar riscos de segurança".

Os riscos identificados deveriam ser ordenados baseados nas prioridades da organização, na probabilidade de ocorrência, na incerteza associada com eles e em seu impacto. A equipe de segurança revisa cada risco, podendo atribuir uma medida de impacto

(alto, médio, ou baixo). Um risco pode ser reduzido (ou mitigado), evitado, transferido ou aceito.

6.3.6. Subprocesso: Especificar Necessidades de Segurança

O propósito do subprocesso Especificar Necessidades de Segurança (Figura 6.9) é especificar as necessidades relacionadas com segurança para o software entre todos os envolvidos (principalmente o usuário), em cada iteração.

Esse subprocesso envolve definir as bases para segurança no software de modo a satisfazer os requisitos legais e organizacionais para segurança. Essas necessidades baseiam-se no contexto de segurança operacional do software, nos objetivos de segurança, no ambiente atual de segurança e sistemas da organização.

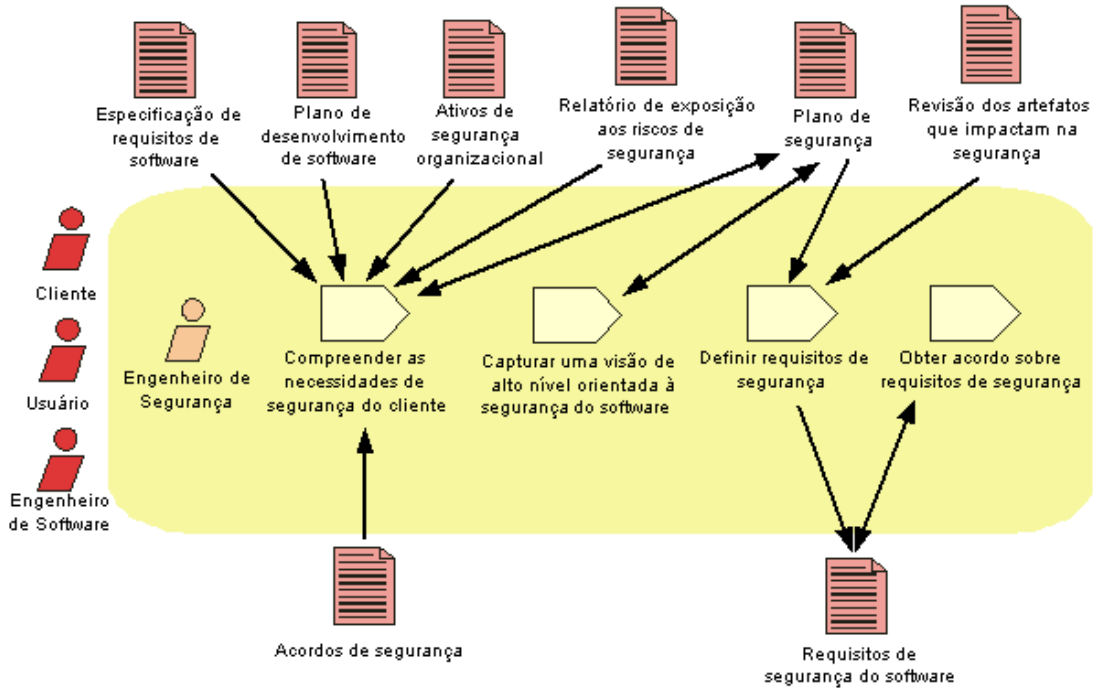

Figura 6.9 - Subprocesso: Especificar Necessidades de Segurança

O Engenheiro de Segurança, com apoio do Usuário e do Cliente, é responsável pela compreensão das necessidades de segurança do cliente, desenvolvendo uma visão de alto nível orientada à segurança do software. A visão de alto nível ajuda na definição dos requisitos de segurança. Finalmente, o Engenheiro de Software faz a mediação para obter acordo sobre esses requisitos de segurança.

6.3.6.1 Artefatos

A Tabela 6.6 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.6: Artefatos do subprocesso "Especificar Necessidades de Segurança"

Artefato	Descrição
Plano de desenvolvimento de software	Ver Tabela 6.1.
Ativos de segurança organizacional	Ver Tabela 6.1.
Plano de segurança	Ver Tabela 6.1.
Acordos de segurança	Ver Tabela 6.1.
Revisão dos artefatos que impactam na segurança	Ver Tabela 6.4.
Relatório de exposição aos riscos de segurança	Ver Tabela 6.5.
Especificação de requisitos de software	Documenta os requisitos de software para o sistema ou para uma parte do sistema.
Requisitos de segurança do software	Documenta os requisitos de segurança identificados através das necessidades de segurança. Este artefato pode fazer parte da especificação de requisitos de software.

Este subprocesso é composto pelas seguintes atividades: (i) Compreender as necessidades de segurança do cliente; (ii) Capturar uma visão de alto nível orientada à segurança do software; (iii) Definir requisitos de segurança; e (iv) Obter acordo sobre requisitos de segurança.

6.3.6.2 Atividade: Compreender as Necessidades de Segurança do Cliente

O Quadro 6.15 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Compreender as Necessidades de Segurança do Cliente</td></tr>
<tr><td colspan="2">Propósito: coletar as informações necessárias para um entendimento das necessidades de segurança do cliente para o software em desenvolvimento.</td></tr>
<tr><td colspan="2">Tarefas:
- Identificar leis, políticas, padrões, restrições e influências externas que se relacionam com o software
- Identificar o propósito do software para determinar seu contexto de segurança
- Restringir acesso à informação</td></tr>
<tr><td>Artefatos de Entrada:
- Ativos de segurança organizacional
- Plano de segurança
- Relatório de exposição aos riscos de segurança
- Plano de desenvolvimento de software
- Especificação de requisitos de software
- Acordos de segurança</td><td>Artefatos Resultantes:
- Plano de segurança (necessidades de segurança especificadas)</td></tr>
<tr><td colspan="2">Ator: Engenheiro de Segurança, Engenheiro de Software, Usuário, Cliente</td></tr>
</table>

Quadro 6.15: Atividade: Compreender as necessidades de segurança do cliente

As necessidades de segurança são influenciadas segundo sua importância para o cliente do risco de segurança do software em desenvolvimento. O ambiente alvo no qual o software será operado também influencia as necessidades de segurança do cliente. Entrevistas com os usuários do software podem ser conduzidas a fim de descobrir quais são suas necessidades de segurança (McGRAW, 2000e). Convém que controles e responsabilidades específicos para atender a estas necessidades sejam definidos e documentados. Esta atividade possui as três tarefas abaixo.

Identificar leis, políticas, padrões, restrições e influências externas que se relacionam com o software

Identificar leis, estatutos, regulamentações, políticas, padrões comerciais e cláusulas contratuais que gerenciam o ambiente alvo do software. Uma determinação da precedência entre políticas locais e globais deveria ser executada. A política de segurança deve também ser levada em consideração. Entende-se por política de segurança o conjunto de diretrizes, normas, e procedimentos que descrevem critérios de segurança de uma empresa.

Muitos fatores externos à organização também influenciam a variação das necessidades de segurança da organização. Esses fatores incluem a orientação política e mudanças no foco político, desenvolvimentos tecnológicos, influências econômicas, eventos globais, entre outros. Como nenhum destes fatores é estático, eles requerem monitoramento e avaliação periódica do impacto potencial da mudança.

Identificar o propósito do software para determinar seu contexto de segurança

Identificar como o contexto do software impacta a segurança, compreendendo propósito e objetivos do software em termos de estratégia da empresa, do tipo de produto e a intenção de uso. Missão de processamento e cenários de operação são avaliados para considerações de segurança, como dotar o software com balanceamento de cargas ou maior escalabilidade. Uma compreensão de alto nível da ameaça que o software pode sofrer é necessária neste estágio.

Requisitos de performance e funcionais são avaliados em relação a possíveis impactos na segurança. Restrições de operação são também revisadas em relação às suas implicações de segurança. O ambiente pode também incluir interfaces com outras organizações ou sistemas de modo a definir o perímetro de segurança do software. Elementos de interface são determinados os quais ficam tanto dentro quanto fora do perímetro de segurança.

Restringir acesso à informação

A restrição de acesso à informação inicia-se com a classificação da informação. A classificação e seus respectivos controles de proteção devem levar em consideração as necessidades de negócios para compartilhamento ou restrição de informações e os respectivos impactos nos negócios. Convém que informações e resultados de produtos de software que processam dados classificados sejam rotulados de acordo com seu valor e sua sensibilidade para a organização. Em seguida, define-se um conjunto apropriado de procedimentos para rotular e tratar a informação de acordo com o esquema de classificação adotado pela organização, abrangendo os artefatos.

Finalmente, os usuários dos sistemas de aplicação, incluindo o pessoal de suporte, são providos de acesso à informação e às funções dos sistemas de aplicação de acordo com uma política de controle de acesso definida, baseada na classificação da informação e nos requisitos das aplicações individuais do negócio e consistente com a política de acesso à informação organizacional.

6.3.6.3 Atividade: Capturar uma Visão de Alto Nível Orientada à Segurança do Software

O Quadro 6.16 apresenta o detalhamento desta atividade.

Atividade: Capturar uma Visão de Alto Nível Orientada à Segurança do Software	
Propósito: desenvolver uma visão de alto nível orientada à segurança do software, incluindo papéis, responsabilidades, ativos, recursos, proteção pessoal, proteção física e operação.	
Tarefas: - Levantar premissas - Capturar objetivos de alto nível que definem a segurança do software	
Artefatos de Entrada: - Plano de segurança	Artefatos Resultantes: - Plano de segurança (visão de alto nível)
Ator: Engenheiro de Segurança, Engenheiro de Software, Usuário, Cliente	

Quadro 6.16: Atividade: Capturar uma visão de alto nível orientada à segurança do software

Esta visão do software é tipicamente fornecida em um conceito de segurança de operações e deveria incluir uma visão de alto nível de segurança da arquitetura do software, procedimentos, e o ambiente. Esta atividade possui as duas tarefas abaixo.

Levantar premissas

As premissas levantadas para que o software seja considerado seguro podem envolver que o ambiente físico tenha controle de acesso adequado, que o sistema operacional e a camada de rede sejam seguros, ou que apenas determinados administradores operem o

sistema. Essas premissas podem ser utilizadas para atender a algumas das necessidades de segurança ou para alterá-las.

Capturar objetivos de alto nível que definem a segurança do software

Identifica-se que necessidades de segurança deveriam ser atendidas para prover a segurança adequada para o software em seu ambiente operacional. Os objetivos capturados mostram como as ameaças e os riscos identificados anteriormente serão tratados, e como o contexto de segurança será satisfeito. Estes objetivos tipicamente tratam de assuntos como identificação e autenticação de usuário, trilhas de auditoria, características administrativas para implementar e gerenciar controles, instalação e configuração de protótipos, e controles de utilização de recursos para prevenir ataques.

6.3.6.4 Atividade: Definir Requisitos de Segurança

O Quadro 6.17 apresenta o detalhamento desta atividade.

Atividade: Definir Requisitos de Segurança	
Propósito: definir um conjunto consistente de requisitos que estabelecem o nível de segurança a ser implementado no software.	
Tarefa: - Definir requisitos de segurança	
Artefatos de Entrada: - Revisão dos artefatos que impactam na segurança - Plano de segurança	Artefatos Resultantes: - Requisitos de segurança do software
Ator: Engenheiro de Segurança, Engenheiro de Software, Usuário, Cliente	

Quadro 6.17: Atividade: Definir requisitos de segurança

Esta atividade possui a tarefa "Definir requisitos de segurança".

Os artefatos importantes de um software devem ser protegidos contra perda, destruição e falsificação, e categorizados em tipos de registros, tais como contábeis, base de dados, transações, auditoria e procedimentos operacionais, cada qual com detalhes do período de retenção.

É necessário garantir que cada requisito seja consistente com a política, leis, padrões, necessidades de segurança, e restrições no software. A definição de requisitos deve abordar aspectos que permitam ao software detectar falhas ou comportamentos não esperados e gerenciá-los, fornecendo uma base para avaliar a segurança do software em seu ambiente alvo.

Deve-se garantir que pelo menos os seguintes aspectos sejam considerados: requisitos de desempenho, recuperação de erros, procedimentos de reinicialização, procedimentos de contingência, concordância sobre controles utilizados, impacto do novo software ou do

software modificado na segurança da informação da empresa, desenvolvimento do novo software ou manutenção/atualização de software existente não afeta de forma adversa sistemas existentes, o software tem aderência a leis e normas da empresa.

A parte 2 da ISO/IEC 15408 (2005b) apresenta um conjunto variado de requisitos funcionais de segurança que podem ser selecionados para satisfazer às necessidades dos interessados.

6.3.6.5 Atividade: Obter Acordo sobre Requisitos de Segurança

O Quadro 6.18 apresenta o detalhamento desta atividade.

Atividade: Obter Acordo sobre Requisitos de Segurança	
Propósito: obter acordo a respeito de que os requisitos de segurança identificados envolvem as necessidades do cliente.	
Tarefa: - Obter acordo sobre requisitos de segurança	
Artefatos de Entrada: - Requisitos de segurança do software	Artefatos Resultantes: - Requisitos de segurança do software (requisitos aprovados)
Ator: Engenheiro de Segurança, Engenheiro de Software, Usuário, Cliente	

Quadro 6.18: Atividade: Obter acordo sobre requisitos de segurança

Esta atividade possui a tarefa "Obter acordo sobre requisitos de segurança".

Seu objetivo é obter acordo entre todas as partes aplicáveis sobre os requisitos de segurança. Para isso, são realizadas revisões para identificar não conformidades em relação à especificação de requisitos de segurança.

6.3.7. Subprocesso: Fornecer Informação de Segurança

O propósito do subprocesso Fornecer Informação de Segurança (Figura 6.10) é prover aos arquitetos de software, projetistas, implementadores, ou usuários informações de segurança de que eles necessitam.

Esse subprocesso procura gerar informação a fim de que: o máximo de problemas do software sejam revisados em relação a implicações de segurança e sejam resolvidos de acordo com objetivos de segurança; os membros da equipe de projeto compreendam o processo de apoio e entendam aspectos da segurança da informação, para que possam realizar suas funções; e a solução reflita a informação de segurança fornecida.

O Engenheiro de Segurança atua como "*mentoring*" de segurança para os demais componente da equipe de projeto.

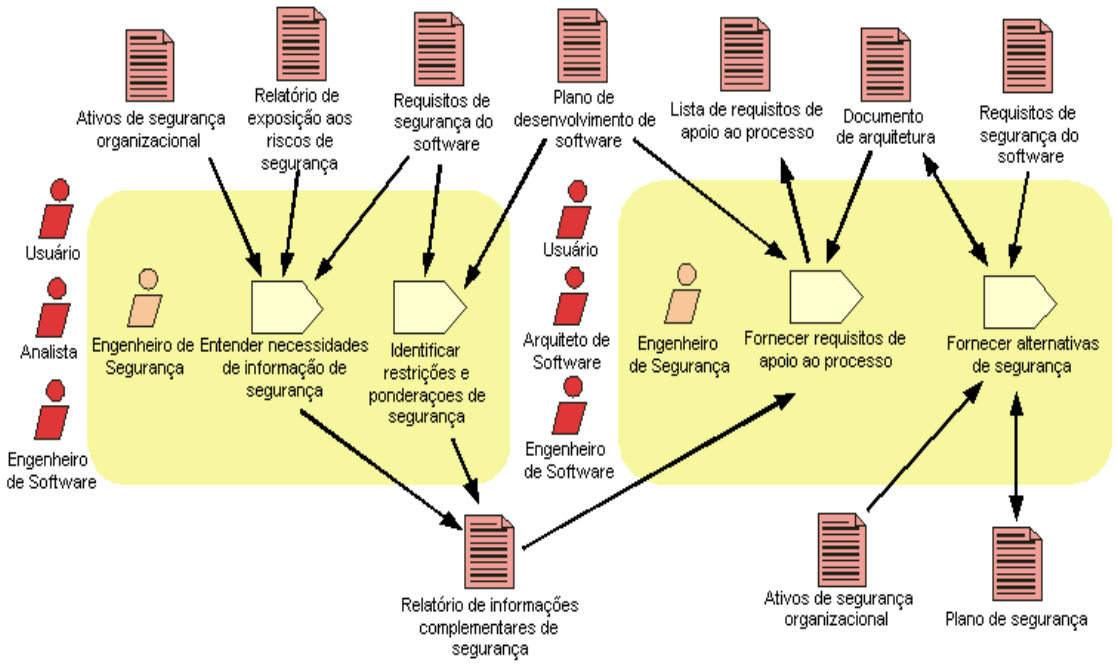

Figura 6.10 - Subprocesso: Fornecer Informação de Segurança

6.3.7.1 Artefatos

A Tabela 6.7 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.7: Artefatos do subprocesso "Fornecer Informação de Segurança"

Artefato	Descrição
Plano de desenvolvimento de software	Ver Tabela 6.1.
Ativos de segurança organizacional	Ver Tabela 6.1.
Plano de segurança	Ver Tabela 6.1.
Documento de arquitetura	Ver Tabela 6.1.
Relatório de exposição aos riscos de segurança	Ver Tabela 6.5.
Requisitos de segurança do software	Ver Tabela 6.6.
Relatório de informações complementares de segurança	Documenta o entendimento das necessidades de segurança através dos requisitos de segurança e outras fontes de informação, relacionando restrições e ponderações de segurança.
Lista de requisitos de apoio ao processo	Documenta os requisitos de apoio ao processo identificados junto aos processos de gerência de configuração e controle de versão, entrega e operação do produto, e documentação.

Este subprocesso é composto pelas seguintes atividades: (i) Entender necessidades de informação de segurança; (ii) Identificar restrições e ponderações de segurança; (iii) Fornecer requisitos de apoio ao processo; e (iv) Fornecer alternativas de segurança.

6.3.7.2 Atividade: Entender Necessidades de Informação de Segurança

O Quadro 6.19 apresenta o detalhamento desta atividade.

Atividade: Entender Necessidades de Informação de Segurança	
Propósito: capturar de projetistas, desenvolvedores, e usuários um entendimento comum das necessidades de informação de segurança de software.	
Tarefa: - Entender necessidades de informação de segurança	
Artefatos de Entrada: - Ativos de segurança organizacional - Requisitos de segurança do software - Relatório de exposição aos riscos de segurança	Artefatos Resultantes: - Relatório de informações complementares de segurança
Ator: Engenheiro de Segurança e Engenheiro de Software	

Quadro 6.19: Atividade: Entender necessidades de informação de segurança

Esta atividade possui a tarefa "Entender necessidades de informação de segurança" que é baseada nas necessidades determinadas no subprocesso "Especificar necessidades de segurança".

6.3.7.3 Atividade: Identificar Restrições e Ponderações de Segurança

O Quadro 6.20 apresenta o detalhamento desta atividade.

Atividade: Identificar Restrições e Ponderações de Segurança	
Propósito: identificar restrições e ponderações de segurança necessárias para tomar decisões do processo e para realizar escolhas de segurança de maneira consistente.	
Tarefa: - Identificar restrições e ponderações de segurança	
Artefatos de Entrada: - Plano de desenvolvimento de software - Requisitos de segurança do software	Artefatos Resultantes: - Relatório de informações complementares de segurança
Ator: Engenheiro de Segurança e Engenheiro de Software	

Quadro 6.20: Atividade: Identificar restrições e ponderações de segurança

Esta atividade possui a tarefa "Identificar restrições e ponderações de segurança".

A equipe de segurança identifica e realiza análises para determinar premissas, restrições e ponderações de segurança em requisitos, projeto, implementação, configuração e documentação, ao longo do ciclo de vida do software. Essas restrições podem ser tanto positivas (sempre fazer algo) ou negativas (nunca fazer algo).

6.3.7.4 Atividade: Fornecer Requisitos de Apoio ao Processo

O Quadro 6.21 apresenta o detalhamento desta atividade.

Atividade: Fornecer Requisitos de Apoio ao Processo	
Propósito: apresentar e analisar os requisitos de apoio a serem implementados, segundo os níveis de segurança que os usuários desejam para o produto.	
Tarefa: - Fornecer requisitos de apoio ao processo	
Artefatos de Entrada: - Plano de desenvolvimento de software - Relatório de informações complementares de segurança - Documento de arquitetura	Artefatos Resultantes: - Lista de requisitos de apoio ao processo
Ator: Engenheiro de Segurança, Engenheiro de Software, Arquiteto de Software, Usuário	

Quadro 6.21: Atividade: Fornecer requisitos de apoio ao processo

Esta atividade possui a tarefa "Fornecer requisitos de apoio ao processo" que define o escopo, o rigor e a profundidade das verificações que irão nortear a análise e a implementação dos requisitos de apoio ao processo.

Esses requisitos de apoio podem estar baseados nos requisitos de garantia de segurança ISO/IEC 15408 (2005c), parte 3, incluindo os requisitos: Gerência de configuração e controle de versão (Classe ACM), Entrega e operação do produto (Classe ADO), e Documentação (Classe AGD).

6.3.7.5 Atividade: Fornecer Alternativas de Segurança

O Quadro 6.22 apresenta o detalhamento desta atividade.

Atividade: Fornecer Alternativas de Segurança	
Propósito: identificar e fornecer soluções alternativas para problemas relacionados com segurança.	
Tarefa: - Fornecer alternativas de segurança	
Artefatos de Entrada: - Ativos de segurança organizacional - Plano de segurança - Requisitos de segurança do software - Documento de arquitetura	Artefatos Resultantes: - Documento de arquitetura - Plano de segurança
Ator: Engenheiro de Segurança, Engenheiro de Software	

Quadro 6.22: Atividade: Fornecer alternativas de segurança

Esta atividade possui a tarefa "Fornecer alternativas de segurança" que objetiva identificar e fornecer soluções alternativas para segurança relacionadas a problemas não identificados no projeto, priorizando-as e criando estratégias de proteção.

Utilizando as restrições e ponderações de segurança identificadas na atividade "Determinar as restrições e ponderações de segurança", engenheiros de segurança podem avaliar cada alternativa de segurança e elaborar uma recomendação para a equipe do projeto.

O Engenheiro de Software deve descrever também as medidas de segurança necessárias para proteger a confidencialidade e a integridade do projeto e da implementação do software no seu ambiente de desenvolvimento. O plano de segurança do projeto deve fornecer evidência de que essas medidas sejam acompanhadas durante o desenvolvimento e a manutenção do software.

6.3.8. Subprocesso: Verificar e Validar Segurança

O propósito do subprocesso Verificar e Validar Segurança (Figura 6.11) é garantir que as soluções sejam verificadas e validadas em relação à segurança ao longo do processo de desenvolvimento.

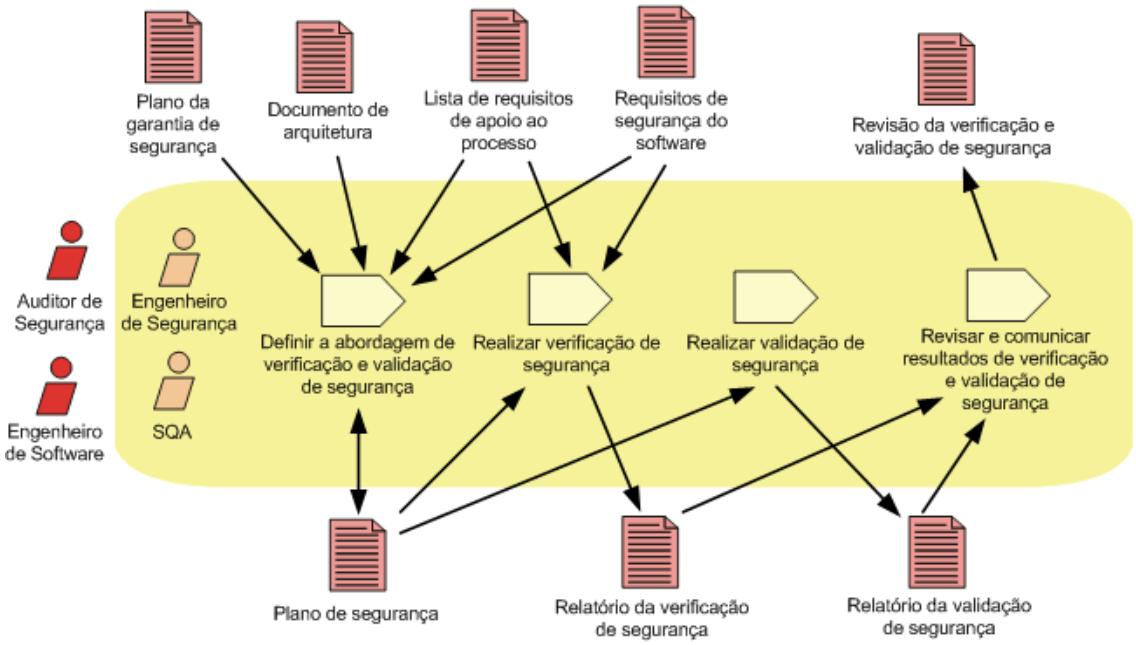

Figura 6.11 - Subprocesso: Verificar e Validar Segurança

Esse subprocesso procura garantir que soluções sejam verificadas em relação a requisitos de segurança, arquitetura, e projeto usando observação, demonstração, análise e teste, também sejam validadas em relação às necessidades de segurança do cliente. Neste contexto, as soluções, de fato, devem implementar de maneira adequada e eficaz os requisitos de segurança, e essas soluções devem satisfazer as necessidades de segurança do cliente.

O Engenheiro de Segurança, sendo auxiliado pelo Engenheiro de Software, define a abordagem de verificação e validação de segurança a qual envolve a elaboração de planos, a cobertura e a profundidade dos testes de verificação e validação. Em seguida, o Engenheiro de Segurança com o apoio do SQA (garantia da qualidade) realiza a verificação e validação de segurança, revisando ao final os resultados e comunicando-os. O Auditor de Segurança acompanha a realização da verificação e da validação a fim de averiguar a correta realização das atividades do subprocesso.

6.3.8.1 Artefatos

A Tabela 6.8 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.8: Artefatos do subprocesso "Verificar e Validar Segurança"

Artefato	Descrição
Documento de arquitetura	Ver Tabela 6.1.
Plano de segurança	Ver Tabela 6.1.
Requisitos de segurança do software	Ver Tabela 6.6.
Lista de requisitos de apoio ao processo	Ver Tabela 6.7.
Plano da garantia de segurança	Documenta os objetivos de garantia de segurança que asseguram um nível mínimo aceitável da segurança implementada no software.
Relatório da verificação de segurança	Documenta os resultados da verificação de segurança e os problemas identificados.
Relatório da validação de segurança	Documenta os resultados da validação de segurança e os problemas identificados.
Revisão da verificação e validação de segurança	Documenta a revisão dos resultados e problemas da verificação e validação de segurança.

Este subprocesso é composto pelas seguintes atividades: (i) Definir a abordagem de verificação e validação de segurança; (ii) Realizar verificação de segurança; (iii) Realizar validação de segurança; e (iv) Revisar e comunicar resultados de verificação e validação de segurança.

6.3.8.2 Atividade: Definir a Abordagem de Verificação e Validação de Segurança

O Quadro 6.23 apresenta o detalhamento desta atividade.

Atividade: Definir a Abordagem de Verificação e Validação de Segurança	
Propósito: definir a abordagem e o nível de rigor para verificar e validar a segurança do software.	
Tarefas: - Elaborar planos de verificação e validação - Definir a cobertura dos testes - Definir a profundidade dos testes	
Artefatos de Entrada: - Documento de arquitetura	Artefatos Resultantes: - Plano de segurança

- Plano de segurança - Lista de requisitos de apoio ao processo - Requisitos de segurança do software - Plano da garantia de segurança	
Ator: Engenheiro de Segurança, Auditor de Segurança, Engenheiro de Software, SQA	

Quadro 6.23: Atividade: Definir a abordagem de verificação e validação de segurança

O objetivo desta atividade é selecionar como os requisitos de segurança devem ser verificados e validados. O nível de rigor deve indicar quão intenso é o exame do esforço da verificação e validação, que é influenciado pela atividade "Definir Estratégia de Garantia de Segurança". A estratégia de garantia é baseada também nos requisitos de segurança aprovados. Se a estratégia de garantia definida for rigorosa, por conseguinte, o nível de rigor da verificação e validação de segurança também será.

Essa atividade inclui também um meio de manter a rastreabilidade das necessidades de segurança do cliente, com os requisitos de garantia de segurança, com as soluções, com os resultados de verificação e validação, entre outros. Esta atividade possui as três tarefas abaixo.

Elaborar planos de verificação e validação

O Engenheiro de Segurança deve elaborar planos de verificação e validação que envolvam a execução de um conjunto de atividades de teste, de análises, observação, e demonstração voltados para descobrir falhas de segurança. Os testes podem ser elaborados através de modelos e padrões de ataque, casos de abuso, e da especificação e projeto. Os testes podem incluir: revisão em pares, teste de invasão, teste de *fuzzing*, aplicar ferramentas de teste, e aplicar ferramenta de análise estática de código.

Definir a cobertura dos testes

Trata da extensão com a qual os testes são realizados e se são ou não suficientes para demonstrar que o aplicativo opera como especificado. O objetivo é estabelecer que as funcionalidades de segurança sejam testadas em relação à sua especificação.

Definir a profundidade dos testes

O teste das funcionalidades de segurança é baseado no aumento da profundidade das informações derivadas das análises. O objetivo é restringir os riscos de não detectar um erro durante o desenvolvimento do produto de software. Além disso, uma vez que o teste é baseado nas estruturas internas do produto, por exemplo, torna-se mais propenso descobrir qualquer código malicioso que foi inserido.

6.3.8.3 Atividade: Realizar Verificação de Segurança

O Quadro 6.24 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Realizar Verificação de Segurança</td></tr>
<tr><td colspan="2">Propósito: verificar que a solução dada implementa os requisitos de segurança.</td></tr>
<tr><td colspan="2">Tarefas:
- Estabelecer critérios de aceitação
- Conduzir testes funcionais de segurança
- Conduzir testes independentes</td></tr>
<tr><td>Artefatos de Entrada:
- Lista de requisitos de apoio ao processo
- Plano de segurança
- Requisitos de segurança do software</td><td>Artefatos Resultantes:
- Relatório da verificação de segurança</td></tr>
<tr><td colspan="2">Ator: Engenheiro de Segurança, Engenheiro de Software, Auditor de Segurança</td></tr>
</table>

Quadro 6.24: Atividade: Realizar verificação de segurança

O objetivo desta atividade é verificar se solução implementa os requisitos de segurança associados ao nível anterior de abstração, incluindo os requisitos de apoio ao processo identificados como resultado da atividade "Fornecer requisitos de apoio ao processo".

Qualquer verificação de conformidade técnica deve ser executada sob supervisão do auditor de segurança. Esta atividade possui as três tarefas abaixo.

Estabelecer critérios de aceitação

São estabelecidos critérios de aceitação para novos protótipos de software, atualizações e novas versões, sendo efetuados testes apropriados dos produtos de software antes da sua aceitação. Esses critérios de aceitação dos protótipos devem estar claramente definidos, acordados, documentados e validados.

Devem considerar: necessidades de desempenho e demanda de capacidade computacional; recuperação de erros, procedimentos de reinicialização e planos de contingência; concordância sobre controles utilizados; procedimentos manuais eficazes; plano de continuidade de negócio; evidência de que a instalação de novo software não afeta de forma adversa sistemas existentes; evidência de que tenha sido considerado o impacto do novo software na segurança da organização; treinamento de novos produtos de software; etc.

Conduzir testes funcionais de segurança

O Engenheiro de Segurança deve conduzir testes funcionais os quais estabelecem que as funcionalidades de segurança exibem as propriedades necessárias para satisfazer os requisitos de segurança do software. Os testes funcionais não estão limitados a resultados positivos que atestam à presença das funções de segurança pedidas; porém, podem também

incluir testes negativos para checar pela ausência de comportamentos indesejados, normalmente baseados na inversão dos requisitos de segurança. Testes de profundidade são também realizados.

Conduzir testes independentes

Especifica que testes podem ser realizados por terceiros com conhecimento especializado e que não participaram do desenvolvimento de software. Seu objetivo é demonstrar que as funções de segurança operam como especificadas. Todavia, há uma necessidade de restringir os riscos de avaliações que não detectam implementações não aderentes às suas especificações.

6.3.8.4 Atividade: Realizar Validação de Segurança

O Quadro 6.25 apresenta o detalhamento desta atividade.

Atividade: Realizar Validação de Segurança	
Propósito: validar a segurança da solução mostrando que ela satisfaz as necessidades associadas com o nível de abstração requerido, bem como as necessidades de segurança do cliente.	
Tarefas: - Validar dados de entrada - Validar controle de processamento interno - Validar dados de saída	
Artefatos de Entrada: - Plano de segurança	Artefatos Resultantes: - Relatório da validação de segurança
Ator: Engenheiro de Segurança, Engenheiro de Software, Auditor de Segurança, SQA	

Quadro 6.25: Atividade: Realizar validação de segurança

O objetivo desta atividade é validar que a solução satisfaça as necessidades associadas com o nível de abstração requerido. A validação demonstra que a solução satisfaz eficazmente estas necessidades. Esta atividade possui as três tarefas abaixo.

Validar dados de entrada

As validações envolvem atestar se os dados de entrada não violam possíveis restrições do software e para garantir que estejam corretos.

Validar controle de processamento interno

Garantia de que algumas restrições e controles são implementados para minimizar o risco de falhas de processamento que possam levar à perda da integridade e confidencialidade. A validação dos controles depende da natureza da aplicação e impacto no negócio.

Validar dados de saída

Os dados de saída de um software são validados para garantir que o processamento e o armazenamento da informação estejam corretos e apropriados.

6.3.8.5 Atividade: Revisar e Comunicar Resultados de Verificação e Validação de Segurança

O Quadro 6.26 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Revisar e Comunicar Resultados de Verificação e Validação de Segurança</td></tr>
<tr><td colspan="2">Propósito: capturar os resultados de verificação e validação de segurança para revisão e comunicar seus resultados aos envolvidos com o projeto de desenvolvimento.</td></tr>
<tr><td colspan="2">Tarefa:
- Revisar e comunicar resultados de verificação e validação de segurança</td></tr>
<tr><td>Artefatos de Entrada:
- Relatório da verificação de segurança
- Relatório da validação de segurança</td><td>Artefatos Resultantes:
- Revisão da verificação e validação de segurança</td></tr>
<tr><td colspan="2">Ator: Engenheiro de Segurança, Engenheiro de Software, Auditor de Segurança, SQA</td></tr>
</table>

Quadro 6.26: Atividade: Revisar e comunicar resultados de verificação e validação de segurança

Esta atividade possui a tarefa "Revisar e comunicar resultados de verificação e validação de segurança" que captura e fornece adequadamente os resultados de verificação e validação.

6.3.9. Subprocesso: Gerenciar Segurança

O propósito do subprocesso Gerenciar Segurança (Figura 6.12) é tratar das tarefas necessárias para administrar e manter os mecanismos de controle de segurança para o ambiente de desenvolvimento, bem como gerenciar a implantação de controles para as funcionalidades de segurança, que se integram com os controles existentes.

Define também como será o gerenciamento da segurança, envolvendo a definição dos treinamentos e programas de educação de segurança necessários. Os serviços de segurança e mecanismos de controle para o projeto de desenvolvimento específico são detalhados.

Outra atribuição desse subprocesso é resolver os problemas identificados com a verificação e validação de segurança.

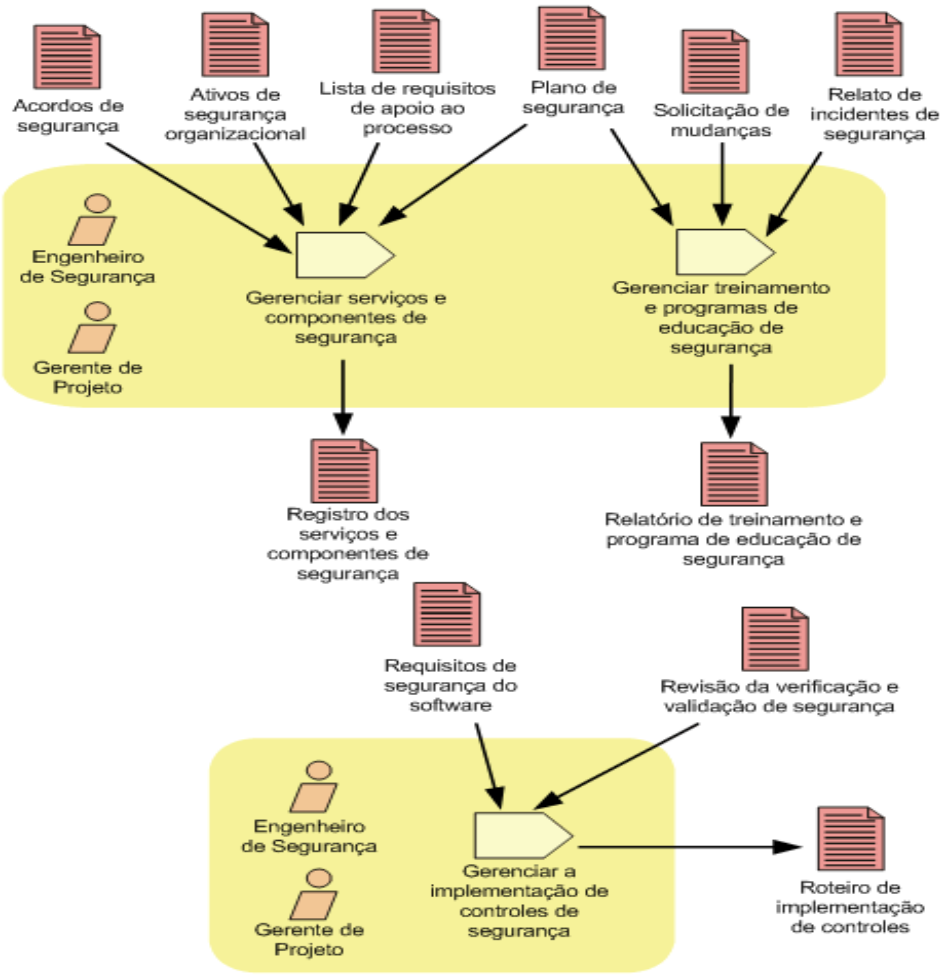

Figura 6.12 - Subprocesso: Gerenciar Segurança

O Engenheiro de Segurança gerencia e controla serviços de apoio e componentes operacionais de segurança, identificando necessidades de treinamento e realizando ações de conscientização sobre o processo de apoio para desenvolver software mais seguro e sobre assuntos de segurança da informação. Gerencia também a implantação de controles de segurança nos produtos de software. O Gerente de Projeto auxilia o Engenheiro de Segurança no gerenciamento destas atividades.

6.3.9.1 Artefatos

A Tabela 6.9 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.9: Artefatos do subprocesso "Gerenciar Segurança"

Artefato	Descrição
Acordos de segurança	Ver Tabela 6.1.
Ativos de segurança organizacional	Ver Tabela 6.1.
Plano de segurança	Ver Tabela 6.1.
Requisitos de segurança do software	Ver Tabela 6.6.

Lista de requisitos de apoio ao processo	Ver Tabela 6.7.
Revisão da verificação e validação de segurança	Ver Tabela 6.8.
Registro dos serviços e componentes de segurança	Documenta os serviços e componentes de segurança que auxiliam o processo de apoio.
Solicitação de mudanças	Usadas para documentar e controlar defeitos, solicitações de melhorias e qualquer outro tipo de solicitação de mudança.
Relato de incidentes de segurança	Documenta informações sobre incidentes de segurança ocorridos.
Relatório de treinamento e programa de educação de segurança	Documenta planos e atividades de treinamento e educação de assuntos relacionados ao processo de apoio.
Roteiro de implementação de controles	Documenta informações sobre a adequação e coordenação da implementação dos controles identificados para a segurança do software a ser desenvolvido.

Este subprocesso é composto pelas seguintes atividades: (i) Gerenciar serviços e componentes de segurança; (ii) Gerenciar treinamento e programas de educação de segurança; e (iii) Gerenciar a implementação de controles de segurança.

6.3.9.2 Atividade: Gerenciar Serviços e Componentes de Segurança

O Quadro 6.27 apresenta o detalhamento desta atividade.

Atividade: Gerenciar Serviços e Componentes de Segurança	
Propósito: gerenciar periodicamente a configuração, manutenção, administração e modificação dos serviços e componentes de segurança.	
Tarefa: - Gerenciar serviços e componentes de segurança	
Artefatos de Entrada: - Acordos de segurança - Ativos de segurança organizacional - Plano de segurança - Lista de requisitos de apoio ao processo	Artefatos Resultantes: - Registro dos serviços e componentes de segurança
Ator: Engenheiro de Segurança, Gerente de Projeto	

Quadro 6.27: Atividade: Gerenciar serviços e componentes de segurança

Esta atividade possui a tarefa "Gerenciar serviços e componentes de segurança".

A configuração, manutenção, administração e modificação dos serviços e componentes (hardware, software e procedimentos) requerem gerenciamento e controle. Isto inclui a proteção contra sua corrupção, tanto acidental quanto deliberada, o estabelecimento dos parâmetros de segurança, e a garantia de atualização dos serviços e componentes.

Faz-se necessário a existência de uma formalização dos procedimentos e das responsabilidades para garantir que haja um controle satisfatório das mudanças de equipamentos, software ou procedimentos.

Modificações no ambiente operacional podem causar impacto em aplicações. Assim, os procedimentos de controle operacional e de aplicações devem ser gerenciados. O controle e o monitoramento inadequados são causas comuns de falha de segurança.

6.3.9.3 Atividade: Gerenciar Treinamento e Programas de Educação de Segurança

O Quadro 6.28 apresenta o detalhamento desta atividade.

Atividade: Gerenciar Treinamento e Programas de Educação de Segurança	
Propósito: gerenciar treinamento e programas de educação de segurança, baseados ou não em alguma percepção, para todos os usuários e administradores do sistema.	
Tarefa: - Gerenciar treinamento e programas de educação de segurança	
Artefatos de Entrada: - Plano de segurança - Solicitação de mudanças - Relato de incidentes de segurança	Artefatos Resultantes: - Relatório de treinamento e programa de educação de segurança
Ator: Engenheiro de Segurança, Gerente de Projeto	

Quadro 6.28: Atividade: Gerenciar treinamento e programas de educação de segurança

Esta atividade possui a tarefa "Gerenciar treinamento e programas de educação de segurança".

O treinamento em segurança auxilia na elaboração de documentação das decisões, inclusive as decisões em contrário, a documentação dos erros e falhas e suas soluções a fim de estimular o aprendizado da organização.

6.3.9.4 Atividade: Gerenciar a Implementação de Controles de Segurança

O Quadro 6.29 apresenta o detalhamento desta atividade.

Atividade: Gerenciar a Implementação de Controles de Segurança	
Propósito: avaliar a adequação e coordenar a implementação de controles específicos de segurança para produtos de software, analisando criticamente os incidentes de segurança previsíveis e resolvendo problemas.	
Tarefas: - Resolver impasses na implementação de controles de segurança - Coordenar decisões e recomendações relacionadas com segurança	
Artefatos de Entrada: - Requisitos de segurança do software - Revisão da verificação e validação de segurança	Artefatos Resultantes: - Roteiro de implementação de controles
Ator: Engenheiro de Segurança, Gerente de Projeto	

Quadro 6.29: Atividade: Gerenciar a implementação de controles de segurança

O objetivo da atividade é avaliar a adequação e coordenar a implementação de controles específicos de segurança para produtos de software em desenvolvimento, analisando criticamente os incidentes de segurança previsíveis.

Um objetivo adicional da atividade é resolver os problemas identificados com a realização do subprocesso "Verificar e Validar Segurança", através da busca pelas causas raiz desses problemas.

Esta atividade possui as duas tarefas abaixo.

Resolver impasses na implementação de controles de segurança

Relacionamentos de sucesso dependem de certas facilidades. Comunicação insuficiente entre diferentes grupos com diferentes prioridades pode resultar em conflitos. É necessário assegurar que disputas sejam resolvidas de maneira apropriada.

Coordenar decisões e recomendações relacionadas com segurança

É necessário comunicar as decisões e recomendações de segurança entre o engenheiro de segurança, equipe de segurança, equipe de sistema, entidades externas, e outras partes apropriadas.

6.3.10. Subprocesso: Garantir Segurança

O propósito do subprocesso Garantir Segurança (Figura 6.13) é definir um conjunto de atividades que possam ser aplicadas para garantir que a segurança do produto foi alcançada e será mantida.

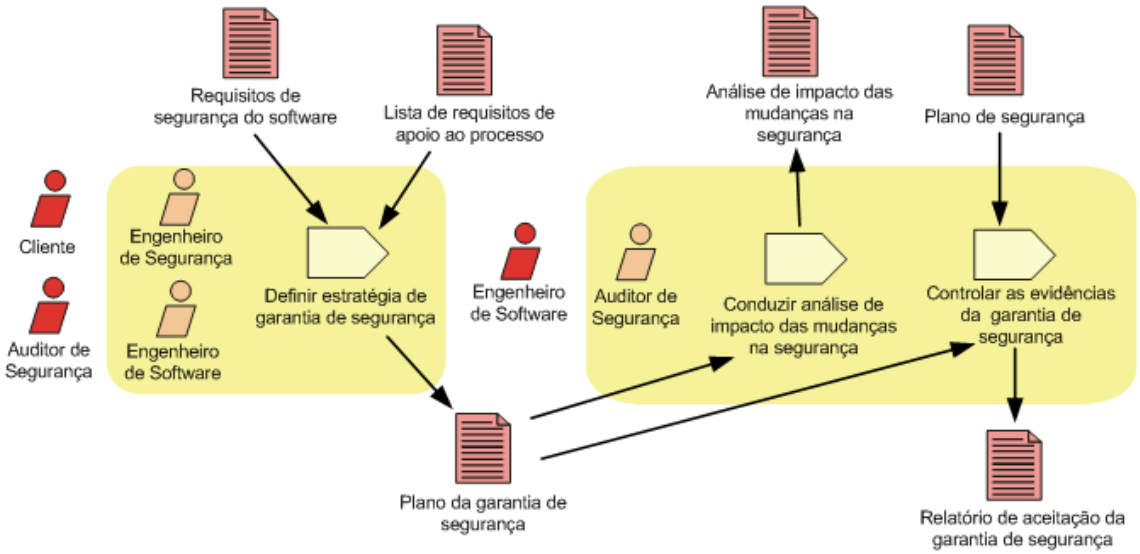

Figura 6.13 - Subprocesso: Garantir Segurança

Esse subprocesso procura assegurar que controles eficazes sejam definidos e implementados para garantir que artefatos (informações e funções) críticos sejam devidamente protegidos.

O Engenheiro de Segurança e o Engenheiro de Software são responsáveis pelo desenvolvimento de uma estratégia de manutenção da garantia de segurança, sendo auxiliados pelo Cliente e pelo Auditor de Segurança. O Auditor de Segurança é responsável pela condução da análise de impacto das mudanças em relação à segurança para atestar que nenhuma mudança comprometa a segurança do software. O Engenheiro de Software e o Auditor de Segurança controlam as evidências da garantia de segurança, que ratificam esta manutenção.

6.3.10.1 Artefatos

A Tabela 6.10 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.10: Artefatos do subprocesso "Garantir Segurança"

Artefato	Descrição
Plano de segurança	Ver Tabela 6.1.
Requisitos de segurança do software	Ver Tabela 6.6.
Lista de requisitos de apoio ao processo	Ver Tabela 6.7.
Plano da garantia de segurança	Ver Tabela 6.8.
Análise de impacto das mudanças na segurança	Documenta os resultados da análise de impacto das mudanças na segurança que afetam o software a ser desenvolvido.
Relatório de aceitação da garantia de segurança	Documenta o resultado da análise das evidências e sua aceitação ou não como forma de ratificar que a garantia de segurança foi satisfeita.

Este subprocesso é composto pelas seguintes atividades: (i) Definir estratégia de garantia de segurança; (ii) Conduzir análise de impacto de segurança das mudanças; e (iii) Controlar as evidências da garantia de segurança.

6.3.10.2 Atividade: Definir Estratégia de Garantia de Segurança

O Quadro 6.30 apresenta o detalhamento desta atividade.

Atividade: Definir Estratégia de Garantia de Segurança.	
Propósito: planejar e definir a estratégia de garantia de segurança de forma a assegurar que o propósito do software e os requisitos de segurança aprovados serão cumpridos e implementados corretamente.	
Tarefas: - Selecionar objetivos de garantia de segurança - Estabelecer plano da garantia de segurança	
Artefatos de Entrada: - Requisitos de segurança do software - Lista de requisitos de apoio ao processo	Artefatos Resultantes: - Plano da garantia de segurança
Ator: Engenheiro de Segurança, Auditor de Segurança, Engenheiro de Software, Cliente	

Quadro 6.30: Atividade: Definir estratégia de garantia de segurança

É importante observar que esta atividade se encerra quando o plano da garantia for aceito pelo Engenheiro de Segurança. Esta atividade possui as duas tarefas abaixo.

Selecionar objetivos de garantia de segurança

A adequação dos objetivos de garantia de segurança é feita pelo engenheiro de software e engenheiro de segurança, a partir dos requisitos de segurança aprovados para o software. Devem ser atualizados para refletir quaisquer mudanças implementadas no produto ou no ambiente.

Estabelecer plano da garantia de segurança

Após a identificação dos objetivos de garantia de segurança, deve ser estabelecido um plano da garantia de segurança que assegure que esses objetivos sejam atendidos. O plano da garantia contém procedimentos que asseguram que a garantia estabelecida seja mantida à medida que mudanças sejam realizadas no software ou em seu ambiente.

6.3.10.3 Atividade: Conduzir Análise de Impacto das Mudanças na Segurança

O Quadro 6.31 apresenta o detalhamento desta atividade.

Atividade: Conduzir Análise de Impacto das Mudanças na Segurança.	
Propósito: analisar o impacto na segurança das mudanças que afetam o software em desenvolvimento.	
Tarefa: - Conduzir análise de impacto das mudanças na segurança	
Artefatos de Entrada: - Plano da garantia de segurança	Artefatos Resultantes: - Análise de impacto das mudanças na segurança
Ator: Auditor de Segurança	

Quadro 6.31: Atividade: Conduzir análise de impacto das mudanças na segurança

Esta atividade possui a tarefa "Conduzir análise de impacto das mudanças na segurança".

O Auditor de Segurança assegura que o plano está sendo seguido por meio da análise de impacto e se o plano é executado corretamente. Isto também é possível através da categorização dos componentes do software de acordo com sua relevância à segurança.

6.3.10.4 Atividade: Controlar as Evidências da Garantia de Segurança

O Quadro 6.32 apresenta o detalhamento desta atividade.

Atividade: Controlar as Evidências da Garantia de Segurança.
Propósito: a confiança de que a garantia de segurança esteja sendo mantida é obtida através da provisão de evidência que ratifica esta garantia de segurança do software.

Tarefas: - Conduzir análise das evidências de segurança - Aceitar formalmente a garantia de segurança	
Artefatos de Entrada: - Plano da garantia de segurança	Artefatos Resultantes: - Relatório de aceitação da garantia de segurança.
Ator: Auditor de Segurança, Engenheiro de Software	

Quadro 6.32: Atividade: Controlar as evidências da garantia de segurança

A equipe de sistema deve fornecer a evidência de que os procedimentos da garantia referenciados no plano estão sendo seguidos, incluindo registros de gerenciamento de configuração, documentação referenciada pela análise de impacto, e evidência do acompanhamento das falhas de segurança, etc. Esta atividade possui as duas tarefas abaixo.

Conduzir análise das evidências de segurança

Esta análise determina se atividades do subprocesso de Verificar e Validar Segurança são suficientemente adequadas e completas para concluir que as características e mecanismos de segurança sejam satisfatoriamente implementados.

Aceitar formalmente a garantia de segurança

A aceitação formal da garantia apoiará a análise da equipe de sistema e a qualidade dessa análise, além de servir para validar o testemunho da equipe de que a garantia foi mantida no software. O auditor de segurança checa se a análise de impacto das mudanças está consistente para a versão atual do software.

6.3.11. Subprocesso: Monitorar Comportamento de Segurança

O propósito do subprocesso Monitorar Comportamento de Segurança (Figura 6.14) é monitorar a segurança do software ou de suas partes liberadas ao longo de sua iteração em seu estado operacional. Isto assegura que deficiências ou erros que poderiam conduzir a uma falha de segurança sejam monitorados e, por conseguinte, identificados, reportados e acompanhados até suas correções.

Neste subprocesso, eventos externos e internos relacionados com segurança são detectados e acompanhados, incidentes são tratados de acordo com a política de segurança, e mudanças na postura de segurança operacional são identificadas e manipuladas de acordo com os objetivos de segurança.

O Engenheiro de Segurança, apoiado pelo Engenheiro de Software, é responsável pelas atividades de monitoração do comportamento de segurança. O Auditor de Segurança acompanha essas atividades com intuito de identificar possíveis não conformidades, reavaliando mudanças nas ameaças, vulnerabilidades, impactos, riscos e ambiente.

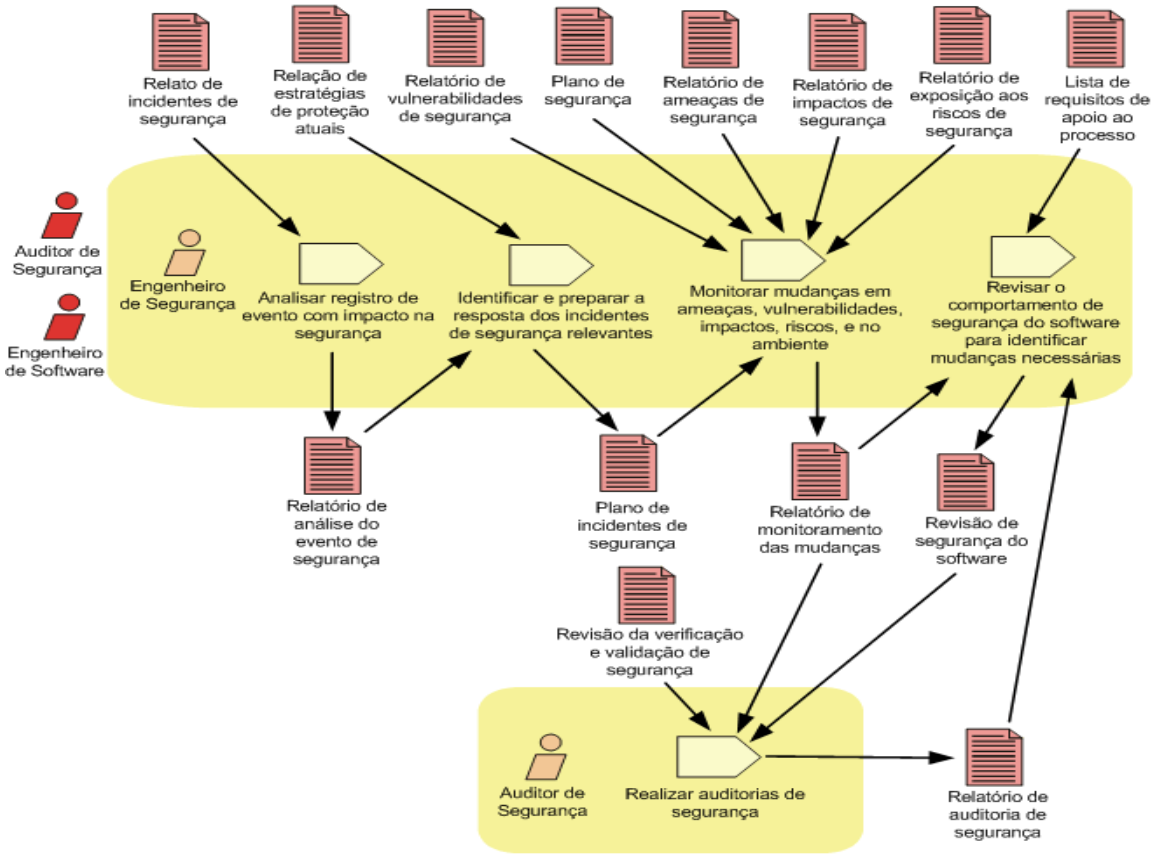

Figura 6.14 - Subprocesso: Monitorar Comportamento de Segurança

6.3.11.1 Artefatos

A Tabela 6.11 apresenta os artefatos sugeridos para este subprocesso.

Tabela 6.11: Artefatos do subprocesso "Monitorar Comportamento de Segurança"

Artefato	Descrição
Plano de segurança	Ver Tabela 6.1.
Relatório de vulnerabilidades de segurança	Ver Tabela 6.2.
Relatório de ameaças de segurança	Ver Tabela 6.3.
Relatório de impactos de segurança	Ver Tabela 6.4.
Relatório de exposição aos riscos de segurança	Ver Tabela 6.5.
Lista de requisitos de apoio ao processo	Ver Tabela 6.7.
Revisão da verificação e validação de segurança	Ver Tabela 6.8.
Relato de incidentes de segurança	Ver Tabela 6.9.
Relatório de análise do evento de segurança	Documenta o resultado da análise dos incidentes de segurança durante a operação do software desenvolvido.
Relação de estratégias de proteção atuais	Documenta informações sobre as proteções de segurança atualmente em uso pela organização e que apóiam a resposta ao incidente.
Plano de incidentes de segurança	Documenta a relação de incidentes de maior

	impacto, as instruções de resposta a esses incidentes, as contingências e o gerenciamento desses incidentes. Pode vir a ser uma seção do plano de segurança.
Relatório de monitoramento das mudanças	Documenta o resultado da monitoração das mudanças consideradas significativas nas ameaças, nas vulnerabilidades, nos impactos, nos riscos, e no ambiente, bem como em suas características.
Revisão de segurança do software	Documenta o exame dos motivos porque a segurança foi aplicada no software desenvolvido, a manutenção da segurança pretendida mesmo após as mudanças, e como os requisitos de apoio ao processo contribuem para esta manutenção.
Relatório de auditoria de segurança	Documenta o resultado da auditoria de segurança, mostrando o resultado da avaliação das mudanças consideradas significativas nas ameaças, nas vulnerabilidades, nos impactos, nos riscos, e no ambiente.

Este subprocesso é composto pelas seguintes atividades: (i) Analisar registro de evento com impacto na segurança; (ii) Identificar e preparar a resposta dos incidentes de segurança relevantes; (iii) Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, e no ambiente; (iv) Revisar o comportamento de segurança do software para identificar mudanças necessárias; e (v) Realizar auditorias de segurança.

6.3.11.2 Atividade: Analisar Registro de Evento com Impacto na Segurança

O Quadro 6.33 apresenta o detalhamento desta atividade.

Atividade: Analisar Registro de Evento com Impacto na Segurança	
Propósito: analisar registros de evento com impacto na segurança para determinar a causa do evento, sua ocorrência, e a probabilidade de eventos futuros.	
Tarefa: - Analisar registro de evento com impacto na segurança	
Artefatos de Entrada: - Relato de incidentes de segurança	Artefatos Resultantes: - Relatório de análise do evento de segurança
Ator: Engenheiro de Segurança, Engenheiro de Software, Auditor de Segurança	

Quadro 6.33: Atividade: Analisar registro de evento com impacto na segurança

Esta atividade possui a tarefa "Analisar registro de evento com impacto na segurança".

O objetivo da atividade é examinar registros de evento (composições de registro de log) e registros históricos para informação de segurança. Os eventos críticos deveriam ser identificados bem como os fatores utilizados para manter a correlação dos eventos entre múltiplos registros. Múltiplos registros de evento podem ser fundidos em um único registro de evento.

6.3.11.3 Atividade: Identificar e Preparar a Resposta dos Incidentes de Segurança Relevantes

O Quadro 6.34 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Identificar e Preparar a Resposta dos Incidentes de Segurança Relevantes</td></tr>
<tr><td colspan="2">Propósito: desenvolver a habilidade de resposta aos incidentes, que causam corrupção em operações do software.</td></tr>
<tr><td colspan="2">Tarefas:
- Compilar informações sobre estratégias atuais de proteção
- Identificar o mau funcionamento de software
- Aprender com os incidentes de segurança</td></tr>
<tr><td>Artefatos de Entrada:
- Relação de estratégias de proteção atuais
- Relatório de análise do evento de segurança</td><td>Artefatos Resultantes:
- Plano de incidentes de segurança</td></tr>
<tr><td colspan="2">Ator: Engenheiro de Segurança, Engenheiro de Software, Auditor de Segurança</td></tr>
</table>

Quadro 6.34: Atividade: Identificar e preparar a resposta dos incidentes de segurança relevantes

O objetivo desta atividade é determinar a ocorrência de um incidente relevante de segurança, identificar os detalhes, e fazer com que o software esteja preparado para responder a este incidente segundo o plano de gerenciamento de incidentes de segurança. O processo pode ser preparado para antecipar incidentes relevantes de segurança usando dados históricos de evento, dados de configuração de software, ferramentas, entre outras informações. Uma resposta de incidente eficaz requer a identificação de um período máximo de funcionamento parcial do software ou a determinação do tempo de inatividade aceitável do software. Quando alguns incidentes ocorrem durante um longo período de tempo, esta análise provavelmente envolverá comparação dos estados do software de tempos em tempos. Esta atividade possui as três tarefas abaixo.

Compilar informações sobre estratégias atuais de proteção

Indicar dentre as práticas de segurança utilizadas pela organização, aquelas que estão sendo seguidas, bem como as que não estão sendo seguidas. Compilar as estratégias de proteção para os processos estratégicos, os processos operacionais, e quaisquer problemas não tratados pela organização.

Identificar o mau funcionamento de software

Convém que um procedimento de identificação do mau funcionamento de software seja estabelecido, junto com um procedimento de resposta ao incidente, estabelecendo a ação a ser tomada ao se receber uma notificação de incidente. Convém que todos os envolvidos estejam conscientes dos procedimentos para identificação, tratamento e recuperação de incidentes de segurança. O mau funcionamento do software pode ser considerado incidente.

Aprender com os incidentes de segurança

Convém que existam mecanismos para permitir que tipos, quantidades e custos dos incidentes e dos maus funcionamentos sejam quantificados, monitorados e armazenados em registros de dados históricos. Esta informação deve ser usada para identificar incidentes ou maus funcionamentos recorrentes ou de alto impacto. Isto pode indicar a necessidade de melhorias ou controles adicionais para limitar a freqüência, danos e custos de ocorrências futuras.

6.3.11.4 Atividade: Monitorar Mudanças em Ameaças, Vulnerabilidades, Impactos, Riscos, e no Ambiente

O Quadro 6.35 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Monitorar Mudanças em Ameaças, Vulnerabilidades, Impactos, Riscos, e no Ambiente</td></tr>
<tr><td colspan="2">Propósito: perceber quaisquer mudanças que possam impactar a eficácia da postura atual de segurança, tanto positiva quanto negativamente.</td></tr>
<tr><td colspan="2">Tarefa:
- Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, e no ambiente</td></tr>
<tr><td>Artefatos de Entrada:
- Plano de segurança
- Relatório de vulnerabilidades de segurança
- Relatório de ameaças de segurança
- Relatório de impactos de segurança
- Relatório de exposição aos riscos de segurança
- Plano de incidentes de segurança</td><td>Artefatos Resultantes:
- Relatório de monitoramento das mudanças</td></tr>
<tr><td colspan="2">Ator: Engenheiro de Segurança, Engenheiro de Software, Auditor de segurança</td></tr>
</table>

Quadro 6.35: Atividade: Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, e no ambiente

Esta atividade possui a tarefa “Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, e no ambiente”.

A segurança implementada para qualquer software deveria ser em relação às ameaças, às vulnerabilidades, aos impactos e aos riscos e da forma como eles se relacionam com os ambientes interno e externo.

Nenhum deles é estático e mudanças influenciam a eficácia e a apropriação de segurança do software. Tudo deve ser monitorado quando ocorre uma mudança, e as mudanças devem ser analisadas para avaliar sua significância em relação à eficácia de segurança.

6.3.11.5 Atividade: Revisar o Comportamento de Segurança do Software para Identificar Mudanças Necessárias

O Quadro 6.36 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Revisar o Comportamento de Segurança do Software para Identificar Mudanças Necessárias.</td></tr>
<tr><td colspan="2">Propósito: reexaminar as razões porque a segurança foi aplicada no software, as necessidades adicionais influenciadas pela segurança aplicadas em outras disciplinas, e os requisitos selecionados de apoio ao processo.</td></tr>
<tr><td colspan="2">Tarefas:
- Examinar procedimento de registro de usuário
- Planejar consumo de recursos
- Monitorar controles de segurança
- Monitorar comportamento do software para identificar brechas de segurança</td></tr>
<tr><td>Artefatos de Entrada:
- Lista de requisitos de apoio ao processo
- Relatório de monitoramento das mudanças
- Relatório de auditoria de segurança</td><td>Artefatos Resultantes:
- Revisão de segurança do software</td></tr>
<tr><td colspan="2">Ator: Engenheiro de Segurança, Engenheiro de Software, Auditor de Segurança</td></tr>
</table>

Quadro 6.36: Atividade: Revisar o comportamento de segurança do software para identificar mudanças necessárias

O comportamento de segurança de um software muda baseado no ambiente de ameaça, nas necessidades operacionais, na configuração do software, e na performance das proteções (controles) de segurança. Esta atividade possui as quatro tarefas abaixo.

Examinar procedimento de registro de usuário

Deve haver um procedimento formal de registro e cancelamento de usuário para obtenção de acesso a todos os sistemas de informação e serviços multiusuários, que exista um controle do acesso através de um processo formal de registro de usuário. Esse procedimento deve tratar a concessão e o uso de privilégios (qualquer característica ou facilidade de um sistema de informação multiusuário, que permita ao usuário sobrepor controles do software ou aplicação), para restringi-los e controlá-los. Finalmente, é necessário que a concessão de senhas seja controlada através de processo de gerenciamento formal e que as senhas sejam armazenadas em sistemas de computador de forma protegida.

Planejar consumo de recursos

Todo programa consome recursos para seus próprios propósitos (cf. seção 4.3). Dentro de certas condições, é possível que o programa comece um ciclo de consumo que termine em instabilidade ou o programa corrompido. Estes problemas causam o que se conhece como "Negação de serviço" (DoS – *Denial of Service*). Para evitar as condições de DoS é uma boa

prática limitar a quantidade de recursos usados, e rejeitar pedidos quando um nível especificado de consumo de recursos for alcançado. Além disso, as demandas de capacidade devem ser monitoradas e as projeções de cargas de produção futuras devem ser feitas de forma a garantir a disponibilidade da capacidade adequada. Convém que essas projeções levem em consideração os requisitos de novos negócios e sistemas e as tendências atuais e projetadas do processamento de informação da organização.

Monitorar controles de segurança

Monitorar e examinar periodicamente a eficácia funcional e a performance dos controles de segurança para identificar possíveis mudanças, fornecendo indicações de seu estado atual, eficácia e necessidades de manutenção. Os controles aplicados nos ambientes operacional e de desenvolvimento devem ser monitorados, pois muitos controles poderiam ser deixados em um estado inapropriado ou não eficaz após período de uso.

Monitorar comportamento do software para identificar brechas de segurança

A equipe de segurança monitora o comportamento do software, rastreando modelos de ameaça e padrões de ataque. O conhecimento adquirido com o entendimento dos ataques, brechas e falhas operacionais deve ser compartilhado com a equipe de sistema e implantado ao longo do ciclo de vida.

6.3.11.6 Atividade: Realizar Auditorias de Segurança

O Quadro 6.37 apresenta o detalhamento desta atividade.

<table>
<tr><td colspan="2">Atividade: Realizar Auditorias de Segurança.</td></tr>
<tr><td colspan="2">Propósito: avaliar se a segurança prevista nas atividades do processo de apoio é alcançada pelo software resultante em seu estado operacional.</td></tr>
<tr><td colspan="2">Tarefa:
- Avaliar mudanças em ameaças, vulnerabilidades, impactos, riscos, e no ambiente</td></tr>
<tr><td>Artefatos de Entrada:
- Revisão de verificação e validação de segurança
- Revisão da verificação e validação de segurança
- Relatório de monitoramento das mudanças
- Revisão de segurança do software</td><td>Artefatos Resultantes:
- Relatório de auditória de segurança</td></tr>
<tr><td colspan="2">Ator: Auditor de Segurança</td></tr>
</table>

Quadro 6.37: Atividade: Realizar auditorias de segurança

Esta atividade trata da realização de auditoria dos produtos das atividades de monitoramento. O auditor de segurança, e opcionalmente uma equipe independente, avalia a aderência do processo de apoio à segurança de software, revisando artefatos do processo de

desenvolvimento. Avalia também mudanças em ameaças, vulnerabilidades, impactos, riscos e ambientes de processamento.

Esta atividade possui a tarefa abaixo.

Avaliar Mudanças em Ameaças, Vulnerabilidades, Impactos, Riscos, e no Ambiente

É necessário avaliar o monitoramento das mudanças em ameaças, vulnerabilidades, impactos, riscos, e no ambiente antes de serem descobertos em produção, podendo colocar clientes e suas informações em perigo. Isto envolve a avaliação de relatórios e a liberação de atualizações e alertas de segurança quando apropriado.

Realiza também a análise *post-mortem* de riscos, ameaças e vulnerabilidades reportados, tomando medidas quando necessário, auxiliando as equipes de sistema e de segurança na adaptação do processo de apoio para que erros similares não sejam repetidos no futuro.

6.4. Conclusão

O Processo de Apoio à Segurança de Software (PASS) foi proposto visando inserir abordagens da engenharia de segurança e da segurança da informação dentro de processos de desenvolvimento de software.

Uma vez que o PASS foi montado a partir de abordagens de segurança já consolidadas e aprovadas pelo mercado, acredita-se que o processo de apoio será uma ferramenta útil e eficaz quando da sua integração com processos padrões de desenvolvimento de software, apoiando empresas a produzirem produtos de software mais seguros.

O capítulo seguinte apresenta uma aplicação do processo de apoio proposto.

Capítulo 7

Aplicação do Processo de Apoio à Segurança de Software

Este capítulo apresenta a aplicação do processo de apoio proposto realizada em uma organização pública.

O Processo de Apoio à Segurança de Software proposto foi especializado para uma organização do setor público brasileiro, seguido de sua aplicação em um projeto piloto.

7.1 Perfil da Organização

A organização do setor público, em que foi aplicado o processo de apoio proposto, é uma instituição de economia mista brasileira, que desenvolve software internamente para dar suporte às suas atividades fins.

A supervisão de desenvolvimento de sistemas dessa organização trabalha com o desenvolvimento e manutenção de soluções de software (Web e Automação de processos). Essas soluções são desenvolvidas obedecendo a uma metodologia proprietária de desenvolvimento de sistemas recentemente elaborada e implantada a qual se baseia no PMBOK (2004) e no RUP (2003). Os projetos desenvolvidos pertencem a diversas áreas, possuem tamanhos variados e são desenvolvidos nas plataformas ASP e J2EE, principalmente.

O principal sistema de informação dessa organização é o seu sistema comercial que realiza o faturamento e a arrecadação mensais de vários serviços prestados. Este sistema ainda funciona na plataforma *mainframe*. Contudo, já se iniciou o projeto para desenvolver um novo sistema comercial em plataforma J2EE. Este novo projeto envolve novas tecnologias, sendo o principal sistema da empresa responsável pelo seu faturamento e pela prestação de serviços. Com a criticidade desse projeto e demais projetos correlatos, o desenvolvimento de sistemas mais seguros passou a ter grande relevância para esta organização, sendo importante a utilização de um processo para desenvolver software mais seguro.

A dinâmica de trabalho para o projeto do novo sistema comercial é em geral operacionalizada como se segue: existe pelo menos um gerente de projeto, um gestor de informática e um gestor da área de negócio, analistas de sistemas e programadores.

O gerente de projetos tem como responsabilidades fazer o acompanhamento gerencial e técnico dos recursos humanos, conduzir reuniões de acompanhamento técnico, negociar alterações de prazos e de escopo com o cliente.

Os gestores de informática e da área de negócio tratam das questões financeiras e quando envolvem mudanças contratuais. Eles são responsáveis por planejar e acompanhar o projeto, tomar as medidas pertinentes para corrigir possíveis desvios encontrados e acompanhar a execução das correções até a sua solução efetiva, alocar pessoal para o papel de gerente do projeto, negociar compromissos com fornecedores e demais gerências da organização.

A metodologia de desenvolvimento de sistemas dessa organização não contemplava processos ou atividades de segurança e nem promovia a utilização de práticas seguras de codificação e desenvolvimento seguro.

7.2 Especialização do Processo de Apoio

O Processo de Apoio à Segurança de Software apresenta 37 atividades alocadas em 11 subprocessos. O PASS foi aplicado em um sistema de segurança e auditoria.

O sistema de segurança e auditoria estava contido no projeto de desenvolvimento do novo sistema comercial da organização. O desenvolvimento desse sistema comercial sofreu atrasos de projeto, repercutindo no prazo para desenvolver o sistema de segurança. Por conseguinte, o prazo para concluir o projeto de desenvolvimento de todo o sistema de segurança foi de 4 semanas. A equipe do projeto do sistema de segurança era composta de apenas 4 membros, o gestor de projeto, 2 analistas (dedicados ao projeto) e o engenheiro de segurança (papel realizado por Francisco José Barreto Nunes).

Para utilizar o PASS no sistema de segurança e auditoria, de forma a manter a restrição de prazo, foram selecionadas as seis primeiras dentre as sete atividades melhor avaliadas em pesquisa de campo realizada com especialistas em processo de desenvolvimento de software e especialistas em segurança da informação, conforme apresentado no capítulo 5. Isso ocorreu porque a última atividade melhor avaliada “Atribuir responsabilidades de segurança no projeto” foi integrada à atividade “Desenvolver plano de segurança”.

Estas atividades são: “Desenvolver plano de segurança”, “Analisar as vulnerabilidades de segurança identificadas”, “Identificar as ameaças de segurança”, “Definir requisitos de segurança”, “Compreender as necessidades de segurança do cliente”, e “Identificar vulnerabilidades de segurança”.

Inicialmente, foi apresentada e explicada cada uma das seis atividades ao gerente de projeto e aos analistas de sistema responsáveis. Avaliou-se conjuntamente a aplicabilidade das seis atividades selecionadas à realidade do projeto com intuito de utilizar aquelas mais adequadas. Ao final, decidiu-se utilizar as seis atividades.

Contudo, alguns artefatos resultantes de cada atividade foram ignorados para reduzir o trabalho adicional que poderia implicar em atraso na entrega do sistema. Os principais artefatos gerados encontram-se nos apêndices deste trabalho.

Em seguida, foi realizada uma apresentação do processo de apoio instanciado para os gestores de negócio, o gestor de informática e o gerente de projeto. O objetivo dessa apresentação foi relatar os benefícios e a importância de se aplicar o processo de apoio no projeto de desenvolvimento, perceber a receptividade e avaliar a aplicabilidade do processo à realidade do projeto. A conclusão foi que o processo poderia ser aplicável neste projeto.

Foi decidido pela organização a não criação do papel do Auditor de Segurança, apesar do reconhecimento de sua importância para as atividades do processo de apoio. O prazo reduzido para a entrega do sistema e a não seleção das atividades de responsabilidade do auditor foram fatores determinantes para a não utilização deste papel.

Uma vez que a organização detém uma metodologia de desenvolvimento de sistema baseada nos conceitos do PMBOK, procurou-se seguir os mesmos mecanismos de coordenação de processos corporativos para o processo de apoio instanciado.

O fluxo do processo de apoio especializado (Figura 7.1) possui os seguintes subprocessos: (*i*) *planejar segurança*; (*ii*) *avaliar vulnerabilidade de segurança*; (*iii*) *modelar ameaça de segurança*; e (*iv*) *especificar necessidades de segurança.*

7.2.1 Planejar Segurança

Este subprocesso foi especializado e inclui a atividade "Desenvolver plano de segurança".

7.2.1.1 Desenvolver Plano de Segurança

Propósito: definir o plano de segurança para o processo de apoio e identificar os mecanismos de se coordenar o processo em um projeto.

Artefatos de entrada utilizados: Plano de desenvolvimento de software.

Artefatos resultantes: Plano de segurança.

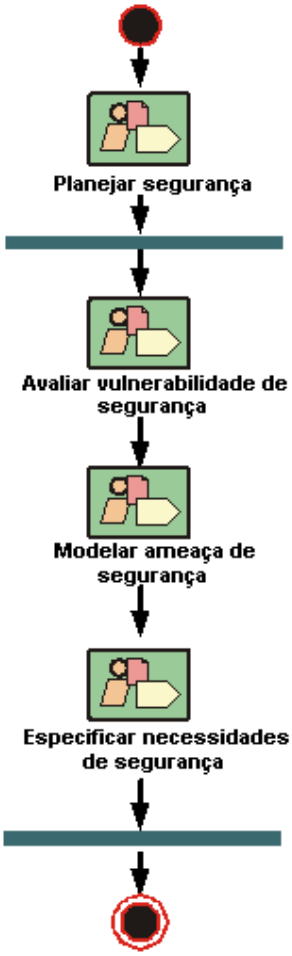

Figura 7.1. Processo de Apoio Especializado

7.2.2 Avaliar Vulnerabilidade de Segurança

Este subprocesso foi especializado e inclui as atividades "Identificar vulnerabilidades de segurança" e "Analisar as vulnerabilidades de segurança identificadas".

7.2.2.1 Identificar Vulnerabilidades de Segurança

Propósito: Executar os métodos, técnicas, e critérios pelos quais as vulnerabilidades de segurança do sistema em um ambiente definido são identificadas e caracterizadas.

Artefatos de entrada utilizados: Plano de desenvolvimento de software, Plano de segurança, Documento de arquitetura.

Artefatos resultantes: Relatório de vulnerabilidades de segurança.

Tarefa realizada: Implementar análise de perigo.

7.2.2.2 Analisar as Vulnerabilidades de Segurança Identificadas

Propósito: As vulnerabilidades de segurança do sistema identificadas são analisadas.

Artefatos de entrada utilizados: Relatório de vulnerabilidades de segurança.

Artefatos resultantes: Relatório de vulnerabilidades de segurança.

Tarefas realizadas: Coletar dados da vulnerabilidade de segurança, Revisar vulnerabilidade de segurança do sistema.

7.2.3 Modelar Ameaça de Segurança

Este subprocesso foi especializado incluindo apenas a atividade "Identificar as ameaças de segurança".

7.2.3.1 Identificar as Ameaças de Segurança

Propósito: Identificar as ameaças de segurança de cada artefato crítico do sistema pelo mapeamento das áreas de perigo para cada artefato crítico com o perfil de ameaça daquele artefato.

Artefatos de entrada utilizados: Relatório de vulnerabilidades de segurança, Plano de segurança.

Artefatos resultantes: Lista de casos de abuso, Lista de árvores de ataque, Relatório de ameaças de segurança.

Tarefas realizadas: Implementar casos de abuso, Implementar árvores de ataque.

7.2.4 Especificar Necessidades de Segurança

Este subprocesso foi especializado incluindo as atividades "Compreender as necessidades de segurança do cliente" e "Definir requisitos de segurança".

7.2.4.1 Compreender as Necessidades de Segurança do Cliente

Propósito: O objetivo desta atividade é coletar as informações necessárias para um entendimento das necessidades de segurança do cliente.

Artefatos de entrada utilizados: Relatório de vulnerabilidades de segurança, Ativos de segurança organizacional, Lista de árvores de ataque, Relatório de ameaças de segurança.

Artefato resultante: Plano de segurança com as necessidades de segurança especificadas.

Tarefas realizadas: Identificar leis, políticas, padrões, restrições e influências externas que se relacionam com o sistema; Identificar o propósito do sistema para determinar seu contexto de segurança.

7.2.4.2 Definir Requisitos de Segurança

Propósito: Definir um conjunto consistente de requisitos que estabelecem o nível de segurança a ser implementado no sistema.

Artefato de entrada utilizado: Plano de segurança.

Artefato resultante: Requisitos de segurança do sistema.

7.3 A Aplicação do Processo de Apoio à Segurança de Software

A aplicação do processo de apoio na organização iniciou-se com um treinamento informal para a equipe de sistema, em que foi apresentado o processo e foram dadas as orientações necessárias à sua utilização.

O projeto piloto utilizado para a instanciação do processo de apoio proposto foi o Sistema de Auditoria e Segurança. Esse sistema foi selecionado devido à sua importância inerente para a segurança dos acessos de todos os usuários dos sistemas corporativos da organização.

O Sistema de Auditoria e Segurança tem como objetivo controlar o acesso de usuários aos demais sistemas da organização através do estabelecimento do perfil no qual o usuário se enquadra. Ele estabelece também padrões para a criação, alteração e manutenção das senhas de forma segura e de acordo com a política de segurança adotada, permitindo a configuração de dados para auditoria.

A maioria do conhecimento das regras de negócio desse sistema originou-se de analistas de sistema da própria organização.

Na aplicação do processo de apoio no projeto piloto selecionado, foram identificados os seguintes pontos relevantes:

a) Além das seis atividades utilizadas inicialmente no processo de apoio especializado, as seguintes atividades foram executadas para se obter um resultado mais satisfatório na aplicação desse processo: "Classificar as ameaças de segurança", "Obter acordo sobre requisitos de segurança", "Realizar verificação de segurança", e "Realizar validação de segurança".
b) Durante a atividade "Desenvolver plano de segurança", foi identificado como documento de segurança para o projeto de desenvolvimento do sistema de auditoria e segurança apenas a norma interna relacionada ao controle de criação e utilização de senha.
c) Pela criticidade inerente de um processo de controle de acesso e auditoria, procurou-se identificar as habilidades da equipe de sistema em codificação segura. Foi constatado o desconhecimento pela maioria dos envolvidos sobre práticas de codificação segura. Logo, foi sugerida e aceita a realização de

treinamento de codificação segura em Java para melhorar a segurança do código escrito para o desenvolvimento do software.

d) Não há ainda na organização uma base histórica de erros e problemas nos sistemas corporativos, que permita identificar com maior brevidade algumas possíveis vulnerabilidades e ameaças de segurança. Em conseqüência disto, gastou-se um tempo considerável para serem identificadas as potenciais vulnerabilidades e ameaças do sistema em estudo.
e) Houve certa dificuldade na realização da tarefa "Implementar análise de perigo" devido a pouca experiência da equipe nesta prática.
f) Não houve necessidade de implementar ações de curto prazo para as vulnerabilidades revisadas, uma vez que o desenvolvimento das regras de negócio do sistema para essas vulnerabilidades ainda não havia iniciado;
g) A tarefa "Identificar o propósito do sistema para determinar seu contexto de segurança" serviu para garantir que as necessidades de segurança coletadas fossem realmente necessárias e justificassem o custo para implantar os controles de segurança destas necessidades.
h) Obedecendo aos padrões da organização, foi utilizado o documento padrão de requisitos para representar em requisitos de segurança as necessidades de segurança entendidas e acordadas.

A análise de perigo foi finalizada com a identificação de alguns problemas e suas vulnerabilidades. Em seguida, as vulnerabilidades identificadas foram detalhadas e encaminhadas ao analista de sistema e ao principal usuário para análise adicional e revisão final, conforme tabela 7.1.

Tabela 7.1: Lista de vulnerabilidades

Vulnerabilidade	Descrição	Criticidade
01. Falha na operação de cadastro dos usuários dos sistemas	A ineficiência no controle para cadastrar os usuários dos sistemas pode ocasionar atribuições incorretas de poder	Médio
02. Falha na operação de cadastro de ações	A ineficiência no controle para cadastrar as ações dos usuários dos sistemas pode ocasionar atribuições incorretas de direito sobre os sistemas	Médio
03. Falha no processo de autenticação	A utilização de critérios de autenticação ineficientes pode causar acessos indevidos e roubo de informações. Aconselha-se usar mecanismos de autenticação padrões de mercado para sistema/funções mais críticos	Alta
04. Falha na função de validação da senha	Permitir que usuários criem senhas de fácil descoberta comprometerá a segurança do acesso aos sistemas	Alta

05. Falha no envio da primeira senha	O não controle do envio da senha inicial pode permitir o envio incorreto de mensagens com a senha inicial gerada pelo sistema	Alta
06. A criptografia da senha não foi realizada no cliente ou a criptografia utilizada é falha	Utilizar algoritmos de criptografia próprios ou que não sejam padrões de mercado pode comprometer a segurança das senhas e demais informações criptografadas	Médio
07. Falha na verificação de acesso	A utilização de controles de acesso ineficientes podem causar acessos indevidos e roubo de informações. Aconselha-se usar mecanismos de acesso padronizados para sistema/funções mais críticos	Alta
08. Falha de segurança na manutenção dos registros de auditoria	As bases e registros de auditoria que não sejam protegidos podem sofrer alterações indevidas para acobertar ações ilegais que ocorram na operação de algum sistema.	Alta

A revisão final das vulnerabilidades mostrou as seguintes coincidências:

- As vulnerabilidades 01 e 05 se complementam, pois no ato do cadastro do usuário, a inserção errada do e-mail pode causar envio errado da senha.
- A vulnerabilidade 03 contempla a vulnerabilidade 07. Isto é, a verificação de acesso é uma das ações do processo de autenticar o usuário. Logo, quando se trata a vulnerabilidade 03 estará tratando a vulnerabilidade 07.

O subprocesso "Modelar ameaça de segurança" foi finalizado com a identificação das ameaças de segurança aos ativos críticos. A implementação de casos de abuso e árvores de ataque auxiliou nessa identificação. Houve dificuldades na elaboração das árvores de ataque e casos de abuso, principalmente. Em geral, o problema deveu-se à falta de prática no preenchimento destes artefatos. Em relação aos casos de abuso, sua elaboração foi mais difícil por causa da quantidade reduzida de exemplos práticos e reais que subsidiassem sua criação. Independentemente desta constatação, estes artefatos facilitaram o entendimento das ameaças pela equipe de sistema e usuários.

Em seguida, verificou-se a probabilidade e a capacidade das ameaças identificadas. Finalmente, a equipe do sistema analisou e aprovou a relação de ameaças. O tratamento das ameaças seguiu a priorização acordada. O caso de abuso e as árvores de ataque identificadas encontram-se representadas nas figuras 7.2, 7.3 e 7.4, respectivamente.

A notação utilizada nas árvores de ataque segue o recomendado em Howard (2002). O objetivo do ataque é o nó raiz. Os caminhos para as folhas são os meios de atingir o objetivo. Existem nós "E" e "OU". O padrão é o nó "OU" e que não precisa ser representado. Caso haja algum nó "E", este deve ser explicitado. Para os nós "E", só se atinge o objetivo quando todos os nós são satisfeitos. A linha tracejada representa menor probabilidade de sucesso no ataque.

Os círculos abaixo dos nós com menor probabilidade na árvore delineiam porque a ameaça é atenuada.

Figura 7.2: Caso de abuso do sistema de auditoria e segurança

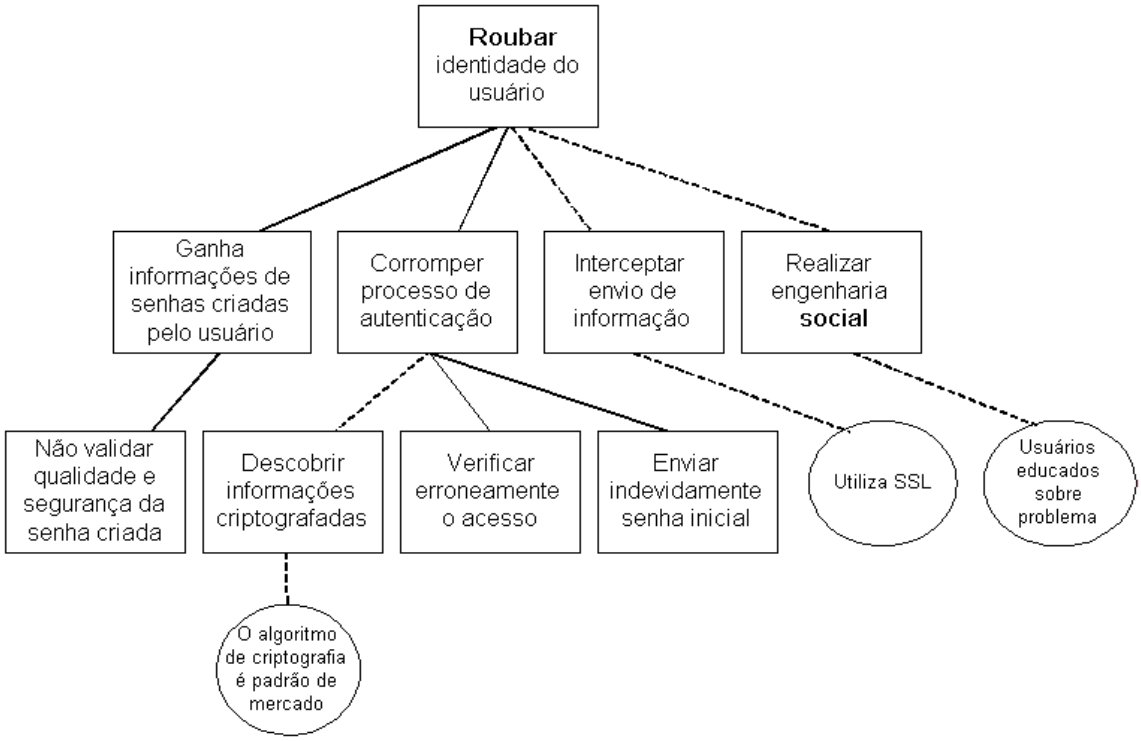

Figura 7.3: Árvore de ataque "Roubar identidade do usuário"

O entendimento das necessidades de segurança ocorreu com o apoio do analista de sistema e engenheiro de segurança. Através da realização das tarefas e com apoio dos artefatos de entrada, identificaram-se as seguintes necessidades de segurança para as ameaças prioritárias:

- A criação automática de senhas de acesso aos sistemas deve obedecer à norma interna de criação de senhas (descritas em manual).
- Deve ser garantido que o acesso ao sistema ocorra de forma segura e que as informações dos usuários não sejam suprimidas.

- Sob hipótese nenhuma o sistema deve permitir que usuários cadastrados, inclusive o administrador, façam alterações na base de registro (log) de auditoria.
- O sistema deve impedir a criação de senhas fáceis e inseguras.

Os requisitos de segurança foram organizados segundo as necessidades de segurança e com apoio das árvores de ataque. Estes requisitos foram apresentados à equipe de sistema para revisão conforme Quadro 7.1.

Figura 7.4: Árvore de ataque "Alterar registro/log de auditoria"

Identificação	Descrição
RSS.01	**Impedir criação de senhas inseguras**
Toda e qualquer criação ou alteração de senha, seja realizada pelo sistema ou pelo usuário, deve seguir, no mínimo, as regras que estabelecem a norma interna de elaboração de senhas. Sempre e quando o usuário compor senhas que não respeitem tais regras, o sistema não permitirá sua criação, solicitando ao usuário a elaboração de outra senha. A criação só será efetivada quando respeitar as regras impostas.	
RSS.02	**Impedir acesso indevido ao sistema**
Os acessos do sistema devem ser seguros com autenticação e validação de perfil a fim de evitar: (*i*) Envio incorreto da senha inicial gerada pelo sistema, (*ii*) Pessoas estranhas acessem algum sistema se passando por usuários válidos, (*iii*) Roubo, alteração ou qualquer outra ação que comprometa as informações dos usuários, (*iv*) O usuário do sistema acessa outras informações além daquelas atribuídas ao seu grupo, (*v*) Cadastro de informações incorretas. Revisões periódicas dos usuários e seus grupos/perfis poderiam reduzir acessos indevidos.	
RSS.03	**Impedir alteração nos registros de auditoria**
Os registros (log) das ações realizadas pelos usuários e pelo administrador do sistema serão armazenados para consulta e estes registros não devem ser alterados ou apagados por estes usuários ou pelo administrador.	

Quadro 7.1. Requisitos de Segurança para Sistema (RSS) de Auditoria e Segurança

Em seguida, foram feitas as melhorias sugeridas nos requisitos de segurança e novamente apresentados para homologação final do documento pelo gerente de projeto, analistas de sistema, engenheiro de segurança, e gestores de negócio responsáveis pelo projeto. A equipe de sistema implementará no sistema a regra de negócio correspondente a cada requisito de segurança homologado.

Em relação ao requisito de segurança RSS.02, já estão previstos controles de segurança para restringir acesso ao sistema por pessoas estranhas, manipular senhas, ou quaisquer ações que comprometam as informações processadas pelo sistema. Estes controles visam evitar problemas como uso incorreto de senhas, *SQL injection*, *Cross-site scripting*, entre outros.

O requisito de segurança RSS.03 está atendido parcialmente, uma vez que foi implementado que qualquer usuário, incluindo o administrador, não poderá alterar ou apagar os registros de auditoria gerados. Inclusive, o sistema só implementa funcionalidades de incluir e consultar tais registros baseados em parâmetros definidos para cada grupo que o usuário pertence. Este requisito só poderá ser considerado completamente atendido após a realização dos testes das regras de negócio que satisfazem tal requisito.

As vulnerabilidades, ameaças e necessidades de segurança relacionadas com menor prioridade deverão ser tratadas após implementação dos requisitos de segurança atualmente identificados.

Posteriormente, foram realizadas entrevistas com a equipe de sistema para obter informações sobre a aplicação do processo de apoio instanciado e sobre as dificuldades encontradas. A seguir, algumas das opiniões sobre a aplicação do processo de apoio instanciado incluem:

- "Considerei o processo de apoio bastante oportuno. Uma sugestão seria manter uma equipe de segurança, o qual aplica o processo de apoio, e com o engenheiro de segurança à frente, independente da equipe de sistema, da condução da metodologia de desenvolvimento organizacional, e da implementação do projeto. A independência da equipe de segurança permite identificar problemas não percebidos pela equipe de sistema.";
- "O processo de apoio revelou-se bastante oportuno face à importância cada vez maior do domínio das informações, que além dos aspectos de eficiência e eficácia, requerem o permanente cuidado quanto à sua segurança.";

- "Foi importante aplicar o processo visto que o sistema em estudo é um sistema de segurança, voltado para Web, e que exige maior proteção desde o lado cliente até o lado servidor.".

A seguir, algumas das dificuldades apontadas na aplicação do processo de apoio instanciado no projeto piloto envolviam:

- Necessitar mais tempo para assimilar melhor a aplicação das atividades do processo de apoio.
- Alinhar o cronograma do processo de desenvolvimento do sistema ao processo de apoio.
- Disponibilizar recursos adicionais para o processo de apoio e dispositivos de integração como forma de garantir a eficiência do processo.
- Identificar os problemas de segurança para montar os cenários de perigo. Isto ocorreu porque os problemas se restringiram apenas a problemas de segurança da aplicação, deixando de tratar questões de segurança de rede (transmissão dos dados), segurança de pessoas e segurança física.

7.4 Conclusão

Este capítulo apresentou a aplicação do Processo de Apoio à Segurança de Software proposto em um projeto piloto de uma organização do setor público.

Apesar da provável complexidade do PASS devido ao grande número de atividades que o compõem, percebe-se ser viável aplicá-lo e que os resultados obtidos com sua aplicação justificariam os recursos adicionais e aumentariam a satisfação dos clientes que almejam software mais seguro.

O próximo capítulo apresenta a conclusão deste trabalho, suas contribuições e trabalhos futuros.

Capítulo 8

Conclusão

Este capítulo apresenta as principais conclusões deste trabalho, suas contribuições e sugestões de trabalhos futuros.

A segurança da informação começou a deixar de ser um mero detalhe e como desenvolvedor somos responsáveis por garantir a segurança dos sistemas que criamos, especialmente hoje que os sistemas estão ligados às redes de computadores. Muitos desenvolvedores sabem da importância de projetar para segurança, mas poucos sabem como incluí-la no processo (McGRAW, 1999).

Clientes estão também mais exigentes em relação a sistemas mais seguros e confiáveis. Os envolvidos com software estão começando a perceber que os sistemas urgem por propriedades emergentes, como: confiabilidade, privacidade, escalabilidade, performance e segurança.

Pode-se recomendar algumas iniciativas básicas para levar as empresas a aplicarem abordagens de segurança no desenvolvimento de seus sistemas, como: (DAVIS, 2004)

- Iniciativas de curto prazo para implementar ações de segurança no processo de desenvolvimento:
    - Incentivar o uso isolado de abordagens e práticas de desenvolvimento de software que possam reduzir defeitos de projeto e implementação de software;
    - Encorajar e apoiar pesquisa para determinar a eficácia dessas abordagens e práticas em reduzir as vulnerabilidades de segurança de software;
    - Definir estratégias para reduzir falhas e vulnerabilidades de segurança de software;
    - Encorajar adoção dessas abordagens e práticas entendidas como úteis para produzir software seguro.

- Iniciativas de médio prazo para implementar ações de segurança no processo de desenvolvimento:
    - Identificar, documentar, e tornar disponível as novas práticas de desenvolvimento seguro que poderiam reduzir vulnerabilidades de segurança de software;
    - Avaliar as práticas que parecem promissoras na produção de software seguro;
    - Analisar os resultados do uso das práticas e abordagens.
- Iniciativas de longo prazo para implementar ações de segurança no processo de desenvolvimento:
    - Organizar as práticas e abordagens em um processo formal agregado à metodologia de desenvolvimento de sistema;
    - Educar e treinar nesse processo formal para desenvolver software seguro;
    - Implantar o processo formal em todos os projetos de desenvolvimento, aperfeiçoando-o sempre.

Como conclusões mais importantes deste trabalho, podem-se destacar:

- A viabilidade e importância de se implantar práticas de segurança no processo de desenvolvimento de software a fim de produzir produtos de software mais confiáveis e seguros.
- A realização da modelagem das ameaças como uma das atividades mais importantes do processo para discernir e ratificar as vulnerabilidades de segurança reais e auxiliar na identificação dos problemas que causariam maior impacto e teriam maior risco. Esta modelagem é realizada através da especificação de casos de abuso, de árvores de ataque ou da implantação de métodos de apoio como o STRIDE ou o DREAD (***D****amage potential*, ***R****eproducibility*, ***E****xploitability*, ***A****ffected users*, e ***D****iscoverability*) (HOWARD, 2002).
- A especificação das necessidades de segurança as quais culminam na definição de requisitos de segurança como uma atividade que conclui a identificação e

seleção dos problemas de segurança mais críticos através da aprovação do cliente e demais envolvidos.

- Ainda não há padrões de segurança para o processo de desenvolvimento de software largamente aceitos.

Podem-se destacar como principais contribuições deste trabalho:

- Uma proposta de um processo de desenvolvimento de software formado por atividades de segurança, permitindo uma visão mais detalhada da segurança de software, pois isto influencia fortemente na produção de software mais seguro.
- Ressaltar a importância de se aplicar boas práticas de segurança da informação no ciclo de vida de desenvolvimento de software.
- Proposição de artefatos de apoio à execução do Processo de Apoio à Segurança de Software.

8.1 Trabalhos Futuros

Dentre os trabalhos futuros que podem dar continuidade à pesquisa aqui apresentada, sugere-se:

- Aplicar o processo de apoio em outro projeto de desenvolvimento de sistemas, porém com a possibilidade de envolver todos os 11 subprocessos e o maior número possível de atividades entre as 37 atividades existentes de acordo com o escopo do projeto.
- Alinhar o Processo de Apoio à Segurança de Software com o RUP, pelo relacionamento dos subprocessos e das atividades com as fases e disciplinas, detalhando em cada atividade do Processo de Apoio, o momento em que haverá a integração com as outras atividades das disciplinas do RUP. Essa proposta permitirá integrar subprocessos e atividades do PASS com fases e disciplinas do RUP. Dessa forma, será possível identificar, por exemplo, como a disciplina de "Teste" e suas atividades relacionam-se com o subprocesso "Verificação e Validação de Segurança" e atividades desse subprocesso.
- Agregar ao processo de apoio proposto padrões de segurança (*security patterns*). Segundo Schumacher (2001), padrões são um conceito para solucionar problemas de projeto de software, sendo uma alternativa para desenvolver software mais seguro. Ou seja, os padrões de segurança descrevem

problemas recorrentes de segurança surgidos em contextos específicos e apresentam um esquema genérico e já provado para solução dos problemas de segurança.

- Aplicar os estudos com o UMLsec (JÜRJENS, 2001; 2002) para aperfeiçoar as atividades propostas nesse trabalho relacionadas à identificação das necessidades de segurança e à definição de requisitos de segurança, possibilitando modelar com mais fidelidade essas necessidades e requisitos de segurança e facilitando sua representação e entendimento entre os envolvidos com o desenvolvimento.
- As métricas de segurança atualmente utilizadas no mercado não oferecem uma imagem completa da postura de segurança do sistema (ARKIN, 2005). Assim, através da contínua aplicação do processo de apoio proposto, pode-se identificar, organizar e utilizar um conjunto de métricas de segurança para medir o progresso e a adequação do sistema aos requisitos de segurança.

8.2 Considerações Finais

Ainda que existam organizações preparadas, que apresentem um bom nível de maturidade e utilizem um processo de desenvolvimento de qualidade, a situação da maioria das empresas de desenvolvimento no que concerne a aplicação de práticas de segurança de software não é das melhores. Segundo Gilliam (2003), o mais crítico é fazer com que a segurança seja integrada ao ciclo de vida do processo de software.

Logo, deveria haver uma política organizacional pró-ativa e viável para garantir a segurança no ciclo de vida que contemplasse desde treinamentos voltados à segurança de software até a inclusão de políticas, padrões, guias, e melhores práticas de segurança no processo de desenvolvimento de sistemas.

Software é usado cada vez mais para substituir processos manuais e, em caso de mal funcionamento, pode ameaçar vidas, saúde, segurança nacional, o ambiente, ou a até mesmo a economia. Esta situação significa que desenvolvedores, patrocinadores, e usuários priorizam software de alta confiança: software que ateste que ele fornece um conjunto de serviços de maneira a satisfazer propriedades críticas específicas (PFLEEGER, 2002). Por essa razão, procura-se ajuda na Engenharia de Software e Engenharia de Segurança para construir software que seja mais seguro e tenha mais qualidade que os desenvolvidos hoje em dia.

Assim, o trabalho realizado buscou a organização e o refinamento de atividades ligadas à engenharia de segurança e à segurança da informação para se montar o Processo de

Apoio à Segurança de Software (PASS). Este processo pretende ajudar equipes de desenvolvimento de software a produzir soluções e sistemas de software com a menor quantidade de problemas de segurança.

Este trabalho abre espaço para realização de novos trabalhos em uma área ainda bastante nova e que necessita de pesquisa e incentivo, com o intuito de fornecer maior suporte para o desenvolvimento de software mais seguro. Um processo que formalize a utilização de um conjunto de atividades de segurança pode ser de grande interesse para grandes empresas, que necessitam de maior segurança nos seus produtos de software e estão dispostas a investir para obter tais recursos. Não obstante, destaca-se que apesar da aplicação de técnicas e métodos de segurança de software não se pode garantir a prevenção de um software sem erro.

REFERÊNCIAS BIBLIOGRÁFICAS

ABRAMS, M. D.; Toth, P. R., 1994, "A Head Start on Assurance: Proceedings of an Invitational Workshop on Information Technology (IT) Assurance and Trustworthiness". NISTIR 5472. Março de 1994.

ALBERTS, C. et al., 2001, "OCTAVE - The Operationally Critical Threat, Asset, and Vulnerability Evaluation". Carnegie Mellon – *Software* Engineering Institute. Disponível em: <www.cert.org/octave>. Acesso em Março de 2007.

ALBERTS, C. et al., 2003, "OCTAVE-S Implementation Guide". Carnegie Mellon – *Software* Engineering Institute. Version 0.9. August 2003. Disponível em: <www.cert.org/octave>. Acesso em Março de 2007.

ALBUQUERQUE, R.; Ribeiro, B., 2002, Segurança no Desenvolvimento de Software. – 2002 - Campus.

ANDRADE, Jeann Marcell Silva. 2005, Avaliação de Processos de Software em Ambientes de Desenvolvimento de Software Orientados à Organização. Dissertação Mestrado em Ciências em Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro (UFRJ). Disponível em: <http://www.cos.ufrj.br/~taba>. Acesso em Maio de 2006.

ARKIN, B.; Stender, S.; McGraw, G., 2005, "Software Penetration Test", IEEE Security and Privacy, January/February 2005, páginas 32-35.

BERARD, B., et al, 2001, System and Software Verification: Model-Checking Techniques and Tools. Springer. 1ª Edição. Agosto de 2001.

CHESS, B., McGraw, G., 2004, "Static Analysis for Security", IEEE Security and Privacy, November/December 2004, páginas 32-34.

CMMI, 2006, Capability Maturity Model Integration, Version 1.2. (CMMI-SE/SW, V1.2 – Continuous Representation), SEI Technical Report CMU/SEI-2006-TR-008.

CC, 2005, Common Criteria, Version 2.3, August 2005. Disponível em: <http://www.commoncriteriaportal.org>. Acesso em Dezembro de 2006.

CEM, 2006, Common Methodology for Information Technology Security Evaluation – Evaluation methodology, Version 3.1. CCMB-2006-09-004. Revision 1. September 2006. Disponível em: <http://www.commoncriteriaportal.org>. Acesso em Dezembro de 2006.

CURPHEY, M. et al., 2002, "A Guide to Building Secure Web Application". The Open Web Application Security Project. Disponível em: <www.owasp.org>. Acesso em Julho de 2006.

DAVIS, Noopur et al., 2004, “Processes to Produce Secure Software: Towards more Secure Software”. Disponível em: <www.cigital.com/papers/download/secure_software_process.pdf>. Acesso em Janeiro de 2006.

DEMING, W. E., 2000, “Out of the Crisis". The MIT Press.

GILLIAM, David P.; Wolfe, Thomas L.; Sherif, Josef S., 2003, "*Software Security Checklist for the Software Life Cycle*". Proceedings of 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE’03).

GREMBA, J.; Myers, C., 1997, “The IDEAL Model: A Practical Guide for Improvement”. SEI Publication – Bridge. Issue 3, 1997. Disponível em: <www.sei.cmu.edu/ideal/ideal.html>. Acesso em Abril de 2006.

HOPE, P.; McGraw, G.; Antón, A. I., 2004, “Misuse and Abuse Cases: Getting Past the Positive”, IEEE Security and Privacy, May/June 2004, páginas 32-34.

HOWARD, M.; LeBlanc D., 2002, Writing Secure Code, 2nd edition. Microsoft Press.

HUMPHREY, Watts S., 1998, “Characterizing the *Software* Process” IEEE *Software*, Volume 5, Nº 2, Março 1998, páginas 73-79.

IEEE Std 610.12, 1990, Standard Glossary of Software Engineering Terminology.

ISO, 2006, International Organization for Standardization. Disponível em <http://www.iso.org>. Acessos diversos de 2006 a 2007

ISO/IEC 15288, 2002, Information technology – Life Cycle Management – System Life Cycle Processes, ISO – International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 15408-1, 2005a, Information technology – Security techniques – Evaluation criteria for IT security – Part 1: Introduction and general model, ISO – International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 15408-2, 2005b, Information technology – Security techniques – Evaluation criteria for IT security – Part 2: Security functional requirements, ISO – International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 15408-3, 2005c, Information technology – Security techniques – Evaluation criteria for IT security – Part 3: Security assurance requirements, ISO – International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 21827, 2002, Information technology – Systems Security Engineering – CMM (SSE-CMM), ISO – International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 27002, 2005, Information technology – Security techniques – Code of practice for information security management, ISO – International Organization for Standardization, Geneva, Switzerland.

JÜRJENS, J., 2001, "*Towards development of secure systems using UMLsec*". In H. Hussmann, editor, Fundamental Approaches to Software Engineering, 4th International Conference, Proceedings, LNCS, pages 187–200. Springer, 2001.

JÜRJENS, J., 2002, "*UMLsec: extending UML for secure systems development*". In UML 2002 - The Unified Modeling Language. Model Engineering, Languages, Concepts, and Tools. 5th International Conference, Dresden, Germany, September/October 2002, Proceedings, volume 2460 of LNCS, pages 412–425. Springer, 2002.

LAVERY, J.; Boldyreff, C., 2001, "Issues in Securing Web-accessible Information Systems". Proceedings of the 10th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE '01).

LIPNER, S.; Howard, M., 2005, "The Trustworthy Computing Security Development Lifecycle". Disponível em: <http://msdn.microsoft.com/library>. Acesso em Janeiro de 2006.

McGRAW, G.; Viega, J., 1999, "Make your *software* behave". Disponível em: <http://www.ibm.com/developerworks/library/s-behave.html>. Acesso em Março de 2007.

McGRAW, G.; Viega, J., 2000a, "Make your *software* behave: Assuring your software is secure". Disponível em: <http://www.ibm.com/developerworks/library/s-assurance.html>. Acesso em Março de 2007.

McGRAW, G.; Viega, J., 2000b, "Software Security Principles: Part 1. The chain is only as strong as its weakest link". Disponível em: <http://www.ibm.com/developerworks/library/s-link.html>. Acesso em Maio de 2007.

McGRAW, G.; Viega, J., 2000c, "Software Security Principles: Part 2. Defense in depth and secure failure". Disponível em: <http://www.ibm.com/developerworks/library/s-fail.html>. Acesso em Maio de 2007.

McGRAW, G.; Viega, J., 2000d, "Software Security Principles: Part 3. Controlling access: Least privilege and compartmentalization". Disponível em: <http://www.ibm.com/developerworks/library/s-priv.html>. Acesso em Maio de 2007.

McGRAW, G.; Viega, J., 2000e, "Software Security Principles: Part 4. Keep it simple, keep it private". Disponível em: <http://www.ibm.com/developerworks/library/s-simp.html>. Acesso em Maio de 2007.

McGRAW, G.; Viega, J., 2000f, "Software Security Principles: Part 5. On keeping secrets, trusting others, and following the crowd". Disponível em: <http://www.ibm.com/developerworks/library/s-princ5.html>. Acesso em Maio de 2007.

McGRAW, G., 2004, "Software Security", IEEE Security and Privacy, March/April 2004, páginas 32-35.

McGRAW, G., 2006, Software security: building security in. Addison-Wesley. 1ª Edição.

MONIN, Jean-François, 2003, Understanding Formal Methods. Springer. 1ª Edição. Janeiro de 2003.

MOOKHEY, K. K., 2005, "Common Criteria: An Overview", Information Systems Control Journal, Volume 1.

PETEANU, Razvan, 2001, "Best Practices for Secure Development", Versão 4.03. Disponível em: <www.cgisecurity.com/lib/best_prac_for_sec_dev4.pdf>. Acesso em Janeiro de 2007.

PFLEEGER, Charles P.; Pfleeger, Shari L., 2002, "Security in Computing". Prentice Hall PTR, 3ª Edição, Dezembro de 2002.

PMBOK Guide 2004 Edition. "A Guide to the Project Management Body of Knowledge", Disponível em <http://www.pmi.org>. Acesso em Dezembro de 2006.

POORE, Ralph S. et al., 1999, "GASSP - Generally Accepted System Security Principles", Versão 2. The International Information Security Foundation (I^2SF). Disponível em: <http://www.infosectoday.com/articles/gassp.pdf>. Acesso em Fevereiro de 2007.

PRIETO-DÍAZ, R., 2004, "Requirements engineering in the information assurance domain: the common criteria evaluation process, perspectives on software requirements", Chapter 7, pages 139 - 168.

RODRIGUEZ, Victor A., 2002, "SPSMM - Secure Programming Standards Methodology Manual", Versão 0.5.1. Disponível em: <http://www.osstmm.org>. Acesso em Março de 2007.

RUP. Rational Unified Process®. Rational Software Corporation. Copyright © 1987 - 2003

SCHNEIER, Bruce, 1999, "Attack Trees: modeling security threats". Dr. Dobbs' Journal. Dezembro de 1999.

SCHUMACHER, M.; Roedig U., 2001, "*Security Engineering with Patterns"*. PLoP 2001 Conference.

SE-CMM, 1995, Systems Engineering Capability Maturity Model, Version 1.1. CMM for Systems Engineering Improvement. CMU/SEI-95-MM-003, SECMM-95-01. November 1995.

SHEARD, Sarah A., Moini, A., 2003, "Security Engineering Awareness for Systems Engineers", Software Productivity Consortium. Published in Proceedings of the 13th Annual Symposium of the International Council on Systems Engineering, Arlington, VA, June-July 2003.

SOMMERVILLE, Ian, 2003, Engenharia de Software, 6ª Edição; Tradução André Maurício de Andrade Ribeiro. São Paulo: Addison Wesley, 2003.

SSE-CMM, 2003, Systems Security Engineering – Capability Maturity Model, Version 3. Disponível em: <http://www.sse-cmm.org>. Acesso em Agosto de 2006.

VALIM, Carlos, E., 2004, A Segurança em Crise – 2004, Revista InformationWeek Brasil, Ano 6, Nº 123, p. 28 – 34, Agosto de 2004.

VERDON, D.; McGraw, G., 2004, “Risk Analysis in Software Design”, IEEE Security and Privacy, May/June 2004, páginas 32-37.

WYK, Kenneth V., 2004, “Working Together for More Secure *Software*”, September 2004. Disponível em: <http://www.esecurityplanet.com/views/article.php/3404191>. Acesso em Janeiro de 2006.

Apêndice 1

MAPEAMENTO DAS ABORDAGENS

SSE-CMM	ISO/IEC 15408	ISO/IEC 27002	OCTAVE
SSE-CMM: AP 05 – Avaliar vulnerabilidade			
PB 05.02 – Identificar vulnerabilidades	3.3.8.1 Análise de Canal de Cobertura 3.3.8.2 Uso Inapropriado 3.3.8.3 Força das Funções de Segurança do Aplicativo	3.2.1.13 Gestão de vulnerabilidades técnicas	3.4.5 (Fase 2)
SSE-CMM : AP 02 – Avaliar impacto			
PB 02.02 – Identificar ativos do sistema	3.3.3 Estabelecer Segurança do Ambiente (Definir ativos que necessitam de proteção)	3.2.1.4 Responsabilidade pelos ativos	3.4.4 Fase 1 (Processo 4: Criar perfis de ameaça) 3.4.4 Fase 2 (Processo 5: Identificar componentes chave)
SSE-CMM : AP 04 - Avaliar ameaça			
PB 04.02 – Identificar ameaças causadas por pessoas	3.3.4 Estabelecer Objetivos de Segurança (Levantar ameaças)	---	3.4.4 Fase 1 (Processo 4: Criar perfis de ameaça)
SSE-CMM : AP 03 - Avaliar risco			
PB 03.03 – Avaliar risco de exposição	---	3.2.1.1 Analisando / avaliando os riscos de segurança da informação	3.4.4 Fase 3 (Processo 7: Conduzir análise de risco)
PB 03.05 – Priorizar riscos	---	3.2.1.2 Tratando os riscos de segurança da informação	3.4.4 Fase 3 (Processo 8: Desenvolver estratégia de proteção)
SSE-CMM : AP 08 - Monitorar postura de segurança			
PB 08.03 - Identificar incidentes de segurança relevantes	---	3.2.1.14 Notificação de fragilidades e eventos de segurança da informação 3.2.1.15 Gestão de incidentes de segurança da informação e melhorias	3.4.4 Fase 3 (Processo 8: Desenvolver estratégia de proteção)
SSE-CMM : AP 10 - Especificar necessidades de segurança			
PB 10.02 – Identificar leis, políticas, padrões, restrições e influências externas que se relacionam com o sistema	3.3.4 Estabelecer Objetivos de Segurança (Levantar políticas de segurança)	3.2.1.16 Conformidade com requisitos legais	---
PB 10.03 – Identificar o propósito do sistema para determinar o contexto de segurança	3.3.3 Estabelecer Segurança do Ambiente (Identificar os objetivos do sistema)	3.2.5.1 Classificação da informação	---
PB 10.04 – Capturar uma visão orientada à segurança da operação do sistema	3.3.4 Estabelecer Objetivos de Segurança (Levantar premissas) 3.3.5 Estabelecer Requisitos de Segurança (Identificar objetivos de segurança)	3.2.1.6 Procedimentos e responsabilidades operacionais 3.2.1.10 Controle de acesso à aplicação e à informação	---
PB 10.06 – Definir um conjunto consistente de declarações que estabelecem a proteção a ser implementada no sistema	3.3.5 Estabelecer Requisitos de Segurança (Especificar requisitos funcionais de segurança)	3.2.1.11 Requisitos de segurança de sistemas de informação	3.4.4 Fase 1 (Processos 1, 2 e 3) 3.4.4 Fase 3 (Processo 8: Desenvolver estratégia de proteção)

SSE-CMM : AP 09 - Fornecer informação de segurança			
PB 09.03 – Identificar soluções alternativas para problemas relacionados com engenharia de segurança	- - -	3.2.1.3 Infra-estrutura da segurança da informação	3.4.4 Fase 3 (Processo 8: Desenvolver estratégia de proteção)
SSE-CMM : AP 11 - Verificar e validar segurança			
PB 11.03 – Verificar que a solução implementa os requisitos associados com o nível anterior de abstração	3.3.7.3 Testes funcionais 3.3.7.4 Testes independentes	3.2.1.7 Planejamento e aceitação dos sistemas 3.2.1.16 Conformidade com requisitos legais 3.2.1.17 Conformidade com normas e políticas de segurança da informação e conformidade técnica	- - -
PB 11.04 – Validar a solução mostrando que ela satisfaz as necessidades associadas com o nível anterior de abstração, bem como as necessidades de segurança operacional do cliente	3.7.3 Evidência da manutenção da garantia de segurança	3.2.1.12 Processamento correto nas aplicações	- - -

Apêndice 2

ASPECTOS QUE PODEM INFLUENCIAR EM UM PROCESSO DE APOIO À SEGURANÇA DE SOFTWARE

1. INTRODUÇÃO

Este questionário objetiva colher a opinião de especialistas em processo de desenvolvimento de software e especialistas em segurança da informação sobre as atividades que devem estar presentes em um processo seguro de desenvolvimento, e os aspectos que podem influenciar a utilização desse processo seguro.

2. CARACTERIZAÇÃO DO ESPECIALISTA

Nome (opcional):		e-mail (opcional):	
ÁREA DE ATUAÇÃO			
Empresa		*Universidade*	
	Consultor de Segurança da Informação		Professor de Engenharia de Software/Computação/Informática
	Security Officer		Professor de Segurança da Informação
	Consultor de implementação de processos		Pesquisador doutor em Engenharia de Software
	Auditor de sistemas/segurança		Pesquisador doutor em Segurança da Informação (ou área relacionada)
	Gerente da Qualidade		Pesquisador doutorando em Engenharia de Software
	Gerente de Projeto		Pesquisador doutorando em Segurança da Informação (ou área relacionada).
	Analista de Sistema		Pesquisador mestrando em Engenharia de Software
	Outro:		Pesquisador mestrando em Segurança da Informação (ou área relacionada).
Tempo de atuação na área: _____ anos		Número de projetos que já participou: _____	
FORMAÇÃO: *Nível e Área*			
	Doutorado	( ) Eng de Software/Computação/Informática ( ) Seg da Informação ( ) Outro	
	Mestrado	( ) Eng de Software/Computação/Informática ( ) Seg da Informação ( ) Outro	
	Especialização	( ) Eng de Software/Computação/Informática ( ) Seg da Informação ( ) Outro	
	Graduação	( ) Eng de Software/Computação/Informática ( ) Seg da Informação ( ) Outro	
EXPERIÊNCIA EM PROCESSO DE SOFTWARE			
Como você classificaria o seu conhecimento da área de processos de software? ( ) Excelente ( ) Alto ( ) Médio ( ) Baixo ( ) Nenhum			
Como você classificaria a sua experiência prática em processos de software? ( ) Excelente ( ) Alto ( ) Médio ( ) Baixo ( ) Nenhum			
EXPERIÊNCIA EM SEGURANÇA DA INFORMAÇÃO			
Como você classificaria o seu conhecimento da área de segurança da informação? ( ) Excelente ( ) Alto ( ) Médio ( ) Baixo ( ) Nenhum			
Como você classificaria a sua experiência prática em segurança da informação? ( ) Excelente ( ) Alto ( ) Médio ( ) Baixo ( ) Nenhum			

3. IMPORTÂNCIA DAS ATIVIDADES PARA O PROCESSO SEGURO

INSTRUÇÕES

Considere o conjunto de atividades listadas abaixo. Marque com um "X" a coluna que representa a sua opinião a respeito do grau de importância de cada atividade para um processo seguro de desenvolvimento, considerando a seguinte escala:

0. Sem importância
1. Pouca importância
2. Média importância
3. Importante
4. Muito importante

O conjunto de atividades é completo? Caso você encontre alguma que não tenha sido incluída no conjunto listado abaixo, acrescente sua descrição no final da tabela, nas linhas em branco e avalie segundo a escala.

Atividades presentes em processo seguro de desenvolvimento:	**0**	**1**	**2**	**3**	**4**
1. Definir objetivos de planejamento de segurança e identificar seus mecanismos					
2. Atribuir responsabilidades de segurança no projeto					
3. Implementar ambientes de processamento					
4. Planejar o gerenciamento de incidentes de segurança					
5. Executar métodos de identificação de vulnerabilidade de segurança					
6. Analisar as vulnerabilidades de segurança identificadas					
7. Identificar as ameaças de segurança aos ativos críticos					
8. Classificar as ameaças de segurança aos ativos					
9. Desenvolver estratégias de redução das ameaças de segurança					
10. Priorizar processos críticos influenciados pelo sistema					
11. Revisar ativos do sistema que se referem à segurança					
12. Identificar e descrever impactos de segurança					
13. Identificar exposição de segurança					
14. Avaliar risco de exposição de segurança					
15. Priorizar riscos de segurança					
16. Compreender as necessidades de segurança do cliente					
17. Capturar uma visão de alto nível orientada à segurança da operação do sistema					
18. Definir requisitos de segurança					
19. Obter acordo sobre requisitos de segurança					
20. Entender e revisar necessidades de informação de segurança					
21. Determinar considerações e restrições de segurança					
22. Identificar e analisar alternativas de segurança					
23. Fornecer orientação de segurança					
24. Identificar e revisar requisitos de garantia de segurança					
25. Definir a abordagem de verificação e validação de segurança					
26. Realizar verificação de segurança					
27. Realizar validação de segurança					
28. Revisar e comunicar resultados de verificação e validação de segurança					
29. Gerenciar e controlar serviços e componentes operacionais de segurança					
30. Gerenciar percepção, treinamento e programa de educação de segurança					
31. Gerenciar a implementação de controles de segurança					
32. Analisar registro de evento com impacto na segurança					
33. Identificar e preparar a resposta dos incidentes de segurança relevantes					
34. Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, no ambiente, e em suas características					
35. Reavaliar mudanças nas ameaças, vulnerabilidades, impactos, riscos e no ambiente					
36. Revisar o comportamento de segurança do sistema para identificar mudanças necessárias					
37. Realizar auditorias de segurança					
38. Definir estratégia de manutenção da garantia de segurança					
39. Conduzir análise de impacto de segurança das mudanças					
40. Controlar as evidências da manutenção da garantia de segurança					

4. ASPECTOS QUE INFLUENCIAM A UTILIZAÇÃO DO PROCESSO SEGURO

INSTRUÇÕES

Considere o conjunto de itens listados abaixo. Marque com um "X" a coluna que representa a sua opinião a respeito do grau no qual um processo seguro de desenvolvimento é influenciado pelos itens listados, considerando a seguinte escala:

0. Sem influência
1. Pouca influência
2. Média influência
3. Influente
4. Muita influência

O conjunto de aspectos é completo? Caso você encontre algum aspecto que não tenha sido incluído no conjunto listado abaixo, acrescente sua descrição no final da tabela, nas linhas em branco e avalie segundo a escala.

Aspectos que podem influenciar a utilização de um processo seguro de desenvolvimento:	**0**	**1**	**2**	**3**	**4**
1. Equipe com experiência em segurança da informação					
2. Gerência com experiência em segurança da informação					
3. Participação do cliente nas etapas do processo					
4. Existência de apoio automatizado					
5. Alto comprometimento da gerência com o projeto					
6. Apoio da direção da empresa					
7. Treinamento formal da equipe no processo seguro					
9. Responsabilidades claramente definidas					
10. Existência de orientações para uso do processo seguro					
11. Apoio à utilização do conhecimento de experiências em projetos anteriores					
12. Ambiente físico de trabalho adequado					
15. Existência de um grupo de Segurança da Informação na empresa					
17. Existência de auditorias da aderência ao processo seguro					
20. Iniciar a implantação do processo seguro por um conjunto específico de atividades					
21. Iniciar a implantação do processo seguro de maneira uniforme					
22. Disciplina na implantação das atividades					
23. O processo seguro estar alinhado aos objetivos de negócio da organização					
24. O processo seguro estar baseado em expectativas realistas					

Apêndice 3

ARTEFATO "PLANO DE SEGURANÇA"

Índice

Plano de Segurança

1. Introdução

[Fornecer uma visão geral do documento, seu objetivo e informe sobre as partes pelas quais ele é composto.]

1.1 Propósito

[Propósito do Plano de Segurança.]

1.2 Escopo

[Especificar qual o escopo do plano, informando a qual projeto se aplica, quais os itens de software cobertos pelo Plano (artefatos, executável, atividade etc).]

1.3 Definições, Acrônimos e Abreviaturas

[Definir todos termos, acrônimos e abreviaturas que venham a causar dúvidas ou interpretações ambíguas neste documento.]

1.4 Documentos de Referência

[Informar os documentos referenciados neste documento, identificando título, a versão do documento, data e responsável pela publicação.]

2. Objetivos de Segurança

[Referenciar o Plano de Desenvolvimento de Software para tratar dos objetivos de segurança do sistema.]

3. Gerenciamento

3.1 Processo de Apoio Especializado

[Descrever a estrutura do Processo de Apoio à Segurança de Software que será aplicada no projeto.]

3.2 Organização

[Descrever a estrutura da organização responsável pela aplicação do processo de apoio.]

3.3 Tarefas

[Apresentar as tarefas do processo de apoio que serão realizadas no projeto, sincronizando com os marcos e os objetivos do projeto.]

3.4 Equipes de Trabalho

[Organizar a formação da equipe de segurança e da equipe de sistemas e identificar as pessoas que comporão estas equipes.]

3.5 Papéis e Responsabilidades

[Identificar papéis e responsabilidades para cada componente de cada equipe em relação à realização das tarefas do processo de apoio.]

4. Documentação do Projeto de Segurança do Sistema

[Relacionar a documentação utilizada e produzida pelo projeto no processo de desenvolvimento. A documentação mínima requerida é composta pelos seguintes artefatos:

- *Ativos de Segurança Organizacional*
- *Acordos de Segurança*
- *Especificação de Requisitos de Software*
- *Especificação de Caso de Uso*
- *Realização de Caso de Uso*
- *Documento de Arquitetura de Software*
- *Plano de Desenvolvimento de Software*
- *Plano de Testes*

- *Plano de Gestão de Configuração*

5. Proposta de Treinamento e Educação de Segurança

[Referenciar as propostas com as necessidades de treinamento e programas de educação relacionados com implantação do processo de apoio e conscientização de segurança da informação.]

6. Descrição dos Ambientes de Processamento

[Relacionar o ambiente de desenvolvimento, de ensaio, e de produção analisados, identificados e implantados.]

7. Plano de Gerenciamento de Incidentes de Segurança

[Definir o plano de incidentes de segurança para o projeto a fim de garantir um resposta rápida, efetiva e ordenada aos incidentes de segurança quando estes ocorrerem.]

8. Perfil das Atividades Críticas

[Referenciar atividades operacionais, de negócio, ou de missão que possuem grande impacto negativo para a organização em caso de comprometimento e que serão tratadas no projeto de desenvolvimento do software.]

9. Especificação das Necessidades de Segurança

[Descrever as necessidades de segurança entendidas pelo cliente para o software a ser desenvolvido.]

10. Visão de Alto Nível da Segurança do Sistema

[Identificar a visão de alto nível desenvolvida para orientar sobre a segurança do software em desenvolvimento.]

11. Abordagem de Verificação e Validação de Segurança

[Definir a abordagem e nível de rigor para verificar e validar a segurança de cada solução ou protótipo de software.]

Apêndice 4

ARTEFATO "RELATÓRIO DE VULNERABILIDADES DE SEGURANÇA"

Índice

Relatório de Vulnerabilidades de Segurança

1. Introdução

[Fornecer uma visão geral do documento, seu objetivo e informe sobre as partes pelas quais ele é composto.]

1.1 Propósito

[Propósito do Relatório de Vulnerabilidades de Segurança.]

1.2 Escopo

[Especificar qual o escopo do relatório, informando a qual projeto se aplica, quais os itens de software cobertos pelo Relatório (artefatos, executável, atividade etc).]

1.3 Definições, Acrônimos e Abreviaturas

[Definir todos termos, acrônimos e abreviaturas que venham a causar dúvidas ou interpretações ambíguas neste documento.]

1.4 Documentos de Referência

[Informar os documentos referenciados neste documento, identificando título, a versão do documento, data e responsável pela publicação.]

2. Método de Identificação de Vulnerabilidades de Segurança

[Referenciar o método, técnicas, e critérios pelos quais as vulnerabilidades de segurança são identificadas e caracterizadas.]

3. Resultados

3.1 Análise de Perigo

[Descrever os estados de perigo do processo de negócio de um sistema, por exemplo: pelo uso inapropriado.] (Opcional)

3.2 Análise da Força das Funções de Segurança

[Descrever se as funcionalidades ou mecanismos de segurança utilizados mantém sua funcionalidade e se são qualificados para atender às necessidades a que são responsáveis.] (Opcional)

3.3 Relação de Vulnerabilidades de Segurança

[Apresentar e caracterizar as vulnerabilidades de segurança identificadas, incluindo o resultado das análises e revisões dessas vulnerabilidades.]

3.4 Lista de Ações de Curto Prazo para Correção das Vulnerabilidades

[Organizar uma lista de ações a serem implementadas em curto prazo para corrigir as vulnerabilidades mais críticas.]

Apêndice 5

ARTEFATO "RELATÓRIO DE AMEAÇAS DE SEGURANÇA"

Índice

Relatório de Ameaças de Segurança

1. Introdução

[Fornecer uma visão geral do documento, seu objetivo e informe sobre as partes pelas quais ele é composto.]

1.1 Propósito

[Propósito do Relatório de Ameaças de Segurança.]

1.2 Escopo

[Especificar qual o escopo do relatório, informando a qual projeto se aplica, quais os itens de software cobertos pelo Relatório (artefatos, executável, atividade etc).]

1.3 Definições, Acrônimos e Abreviaturas

[Definir todos termos, acrônimos e abreviaturas que venham a causar dúvidas ou interpretações ambíguas neste documento.]

1.4 Documentos de Referência

[Informar os documentos referenciados neste documento, identificando título, a versão do documento, data e responsável pela publicação.]

2. Relação de Casos de Abuso

[Referenciar os casos de abuso preparados a partir da análise da iteração do software em desenvolvimento.] (Opcional)

3. Relação de Árvores de Ataque

[Referenciar as árvores de ataque preparadas a partir da análise da iteração do software em desenvolvimento.] (Opcional)

4. Resultados

4.1 Ameaças de Segurança Identificadas

[Identificar as ameaças de segurança para cada artefato (ativo) crítico do software em desenvolvimento.]

4.2 Classificação das Ameaças Identificadas

[Classificar as ameaças identificadas pela sua criticidade, verossimilhança, capacidade do agente da ameaça e pela probabilidade.]

4.3 Estratégias de Redução das Ameaças de Segurança

[Apresentar conjunto de estratégias para reduzir a probabilidade da ocorrência das ameaças.]

Apêndice 6

ARTEFATO "RELATÓRIO DE IMPACTOS DE SEGURANÇA"

Índice

Relatório de Impactos de Segurança

1. Introdução

[Fornecer uma visão geral do documento, seu objetivo e informe sobre as partes pelas quais ele é composto.]

1.1 Propósito

[Propósito do Relatório de Impactos de Segurança.]

1.2 Escopo

[Especificar qual o escopo do relatório, informando a qual projeto se aplica, quais os itens de software cobertos pelo Relatório (artefatos, executável, atividade etc).]

1.3 Definições, Acrônimos e Abreviaturas

[Definir todos termos, acrônimos e abreviaturas que venham a causar dúvidas ou interpretações ambíguas neste documento.]

1.4 Documentos de Referência

[Informar os documentos referenciados neste documento, identificando título, a versão do documento, data e responsável pela publicação.]

2. Priorização das Atividades Críticas para Segurança

[Apresentar resumo das informações do Plano de Segurança sobre as atividades críticas a serem tratadas pelo software em desenvolvimento.]

3. Revisão dos Artefatos que Impactam na Segurança

[Referenciar o resultado da revisão e caracterização dos artefatos (ativos), para a iteração do software em desenvolvimento, que suportam as capacidades chave e/ou os objetivos de segurança do software.] (Opcional)

4. Relação dos Impactos de Segurança

[Identificar e descrever os impactos de segurança não desejados dos incidentes ocorridos em cada artefato (ativo) crítico do software em desenvolvimento.]

Apêndice 7

ARTEFATO "RELATÓRIO DE EXPOSIÇÃO AOS RISCOS DE SEGURANÇA"

Índice

Relatório de Exposição aos Riscos de Segurança

1. Introdução

[Fornecer uma visão geral do documento, seu objetivo e informe sobre as partes pelas quais ele é composto.]

1.1 Propósito

[Propósito do Relatório de Exposição aos Riscos de Segurança.]

1.2 Escopo

[Especificar qual o escopo do relatório, informando a qual projeto se aplica, quais os itens de software cobertos pelo Relatório (artefatos, executável, atividade etc).]

1.3 Definições, Acrônimos e Abreviaturas

[Definir todos termos, acrônimos e abreviaturas que venham a causar dúvidas ou interpretações ambíguas neste documento.]

1.4 Documentos de Referência

[Informar os documentos referenciados neste documento, identificando título, a versão do documento, data e responsável pela publicação.]

2. Identificação da Exposição de Segurança

[Referenciar a exposição de segurança do software. A exposição envolve as ameaças, as vulnerabilidades e os impactos de segurança.]

3. Risco de Exposição de Segurança

[Avaliar os riscos associados com cada exposição de segurança identificada através de sua probabilidade de ocorrência e nível de incerteza.]

4. Plano de Mitigação de Risco de Segurança

[Referenciar o plano de redução do risco de segurança através da proposição de controles de proteção para os artefatos (ativos) críticos do software em desenvolvimento.]

5. Priorização de Riscos de Segurança

[Apresentar os riscos de segurança em ordem de prioridade segundo as prioridades do projeto de desenvolvimento, prioridades da organização, impactos negativos, nível de incerteza, entre outros critérios.]

Livros Grátis

(http://www.livrosgratis.com.br)

Milhares de Livros para Download:

Baixar livros de Literatura
Baixar livros de Literatura de Cordel
Baixar livros de Literatura Infantil
Baixar livros de Matemática
Baixar livros de Medicina
Baixar livros de Medicina Veterinária
Baixar livros de Meio Ambiente
Baixar livros de Meteorologia
Baixar Monografias e TCC
Baixar livros Multidisciplinar
Baixar livros de Música
Baixar livros de Psicologia
Baixar livros de Química
Baixar livros de Saúde Coletiva
Baixar livros de Serviço Social
Baixar livros de Sociologia
Baixar livros de Teologia
Baixar livros de Trabalho
Baixar livros de Turismo