Segurança: Fator Determinante para A Qualidade de Software

Francisco José Barreto Nunes, Arnaldo Dias Belchior

Mestrado em Informática Aplicada - Universidade de Fortaleza (UNIFOR)
Av. Washington Soares, 1321 CEP 60.811-341 Fortaleza – CE – Brasil

fcojbn@yahoo.com.br, belchior@unifor.br

Abstract. *The increasing occurrence of security failures in software products alerts for the low quality of developed software. However, information security has a larger scope that can combine different quality aspects. A solution for this tendency of failures is to include activities usually applied in information security in a software development process. Therefore, this work proposes a process to deal software security quality: the Software Development Secure Process.*

Resumo. *A crescente incidência de falhas de segurança decorrentes de problemas nos produtos de software alerta para a baixa qualidade do software desenvolvido. Contudo, a segurança da informação tem um escopo maior que pode combinar diversos aspectos de qualidade. Uma solução para essa tendência de falhas é incluir atividades normalmente aplicadas em segurança da informação no processo de desenvolvimento de software. Logo, este trabalho propõe um processo para tratar a qualidade da segurança de software: o Processo Seguro de Desenvolvimento de Software.*

1. Introdução

Falhas de segurança em software são um dos principais problemas enfrentados pelos profissionais de segurança [CERT 2005]. Esta realidade tende a crescer, uma vez que muitas empresas ainda não perceberam o quão é importante agregar princípios de segurança em seus processos de desenvolvimento de software. Todavia, a especialização de atividades de um processo de desenvolvimento voltadas para produzir software mais seguro ainda está em fase de maturação no que se refere aos problemas de segurança dos sistemas. O processo CLASP (2005), por exemplo, é uma iniciativa que objetiva aplicar a segurança dentro do processo de desenvolvimento de software.

A segurança da informação discorre e orienta sobre a melhor forma de aumentar a proteção das informações manipuladas por produtos de software. Não obstante, não se pode deduzir que um software ao implementar algoritmos de criptografia para proteger a integridade dos dados pode ser considerado seguro. Na verdade, esse software apenas implementa uma característica de segurança, não podendo, de fato, ser considerado seguro. A criptografia, por exemplo, não protege o software contra ataques de sobrecarga de memória (*buffer overflow*).

A segurança de software não pode ser confundida com software seguro, isto é, funcionalidades e características de segurança em um software não representam que o software seja seguro [McGraw 2004].

Este trabalho objetiva contribuir para a construção de software seguro e, por conseguinte, software de maior qualidade, por meio da aplicação do Processo Seguro de

Desenvolvimento de Software (PSDS) organizado a partir do SSE-CMM (2003), do OCTAVE [Alberts *et al*. 2001], da ISO/IEC 15408 (2005a, 2005b, 2005c), e da ISO/IEC 17799 (2005).

Este trabalho está organizado como se segue. A seção 2 descreve padrões e normas de segurança utilizadas para estruturar as atividades do processo seguro proposto. A seção 3 apresenta uma visão geral do Processo Seguro de Desenvolvimento de Software. A seção 4 discorre sobre a aplicação do processo seguro. A seção 5 apresenta as principais conclusões deste trabalho.

2. Padrões e Normas de Segurança da Informação

O ISSEA (*International Systems Security Engineering Association*) (2003) estendeu o CMM (*Capability Maturity Model*) [Paulk *et al*. 1993] para o SSE-CMM (*System Security Engineering – Capability Maturity Model*) (2003). A versão anterior do SSE-CMM foi adaptada para se tornar a norma ISO/IEC 21827 (2002).

O SSE-CMM objetiva garantir a segurança de um sistema fornecendo formas de traduzir as necessidades de segurança do cliente em um processo de segurança, que gere produtos que satisfaçam tais necessidades.

O OCTAVE (*Operationally Critical Threat, Asset, and Vulnerability Evaluation*) [Alberts *et al*. 2001] é uma técnica estratégica de planejamento e avaliação baseada em riscos para segurança. Aspectos relacionados aos riscos (ativos, ameaças, vulnerabilidades, e impacto organizacional) são tratados, permitindo que uma organização ajuste a estratégia de proteção de seus riscos de segurança.

Os requisitos do OCTAVE são agrupados em um conjunto de critérios. Neste ponto, dois métodos consistentes com os critérios foram desenvolvidos: o OCTAVE e OCTAVE-S. O Método OCTAVE foi projetado para ser aplicado em grandes organizações. O Método OCTAVE-S é voltado para pequenas organizações.

A ISO/IEC 15408 (*Evaluation criteria for Information Technology security*) (2005a, 2005b, 2005c) propõe um conjunto de critérios para avaliar a segurança de produtos de tecnologia da informação. A ISO/IEC 15408 deriva-se do *Common Criteria* (2005).

Tanto o *Common Criteria* quanto a ISO/IEC 15408 apresentam um conjunto de requisitos que devem ser atendidos para atingir níveis de segurança nos sistemas. Os requisitos se dividem em: requisitos funcionais de segurança, e requisitos de garantia da segurança. Os primeiros requisitos atuam como um conjunto mínimo de características de segurança que um software poderia implementar e envolvem desde a privacidade até a auditoria.

Os requisitos de garantia operam como ações a serem realizadas em apoio ao desenvolvimento, para validar e certificar que o software seja seguro por ter conduzido tais ações (requisitos de garantia), que envolvem: Suporte ao ciclo de vida, Testes, Gerência de configuração, e Entrega e operação do produto, etc.

A ISO/IEC 17799 (2005) objetiva preservar a confidencialidade, a integridade e a disponibilidade das informações, através da implementação de controles. Afirma ser essencial para uma organização identificar seus requisitos de segurança. Isto pode ser conseguido pela avaliação de risco dos ativos através da análise das ameaças, das vulnerabilidades, da probabilidade da ameaça se concretizar, além do impacto dessa ameaça na organização.

3. Processo Seguro de Desenvolvimento de Software

A partir dos padrões e das normas apresentados na seção anterior, tendo como principal referência o SSE-CMM (2003), foram identificados um conjunto de macroatividades e atividades que passaram a compor o processo proposto: Processo Seguro de Desenvolvimento de Software (PSDS).

O PSDS (Figura 1) foi elaborado inicialmente tendo como base o ciclo de vida iterativo incremental. Para a utilização de outros ciclos de vida, isto precisaria ser validado. Este processo é constituído de 37 atividades agrupadas em 11 macroatividades de segurança.

As macroatividades e suas atividades elencadas representam uma proposta não exaustiva. A idéia é que cada atividade se adeque às etapas do ciclo de vida de software e que seja aplicada em paralelo com essas etapas.

Neste contexto, é recomendável especializar o processo proposto para a organização que o utilizar, segundo suas necessidades e peculiaridades. Assim sendo, é possível utilizar apenas um subconjunto das macroatividades e ou das atividades propostas, adaptá-las e ou incluir novas macroatividades ou atividades específicas.

O PSDS apresenta dois papéis que apóiam a execução do processo: o *Engenheiro de segurança*, e o *Auditor de segurança*. O Engenheiro de segurança é responsável pela especialização, e instanciação do processo seguro segundo os objetivos do projeto, alinhado com metas e planos da organização, e verificando se os projetos satisfazem seus objetivos de segurança.

O *Auditor de segurança* avalia a aderência do processo seguro nos projetos de software, em termos de resultados das atividades, dos artefatos produzidos, verificando a conformidade das ações dos projetos ao processo estabelecido. Ele valida a eficaz aplicação do processo seguro, podendo reportar-se aos principais envolvidos através de um canal de comunicação independente, para relatar e tratar de não conformidades identificadas. Preferencialmente, o auditor de segurança não deve acumular as responsabilidades do papel desempenhado pela equipe de garantia da qualidade.

As 11 macroatividades do PSDS serão descritas a seguir.

3.1 Planejar segurança

O propósito dessa macroatividade é garantir que as informações necessárias para o planejamento da segurança para o projeto sejam estabelecidas e registradas.

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 seguro).

Essa macroatividade é composta pelas seguintes atividades: (*i*) Desenvolver plano de segurança; (*ii*) Planejar ambientes de processamento; e (*iii*) Planejar o gerenciamento de incidentes de segurança.

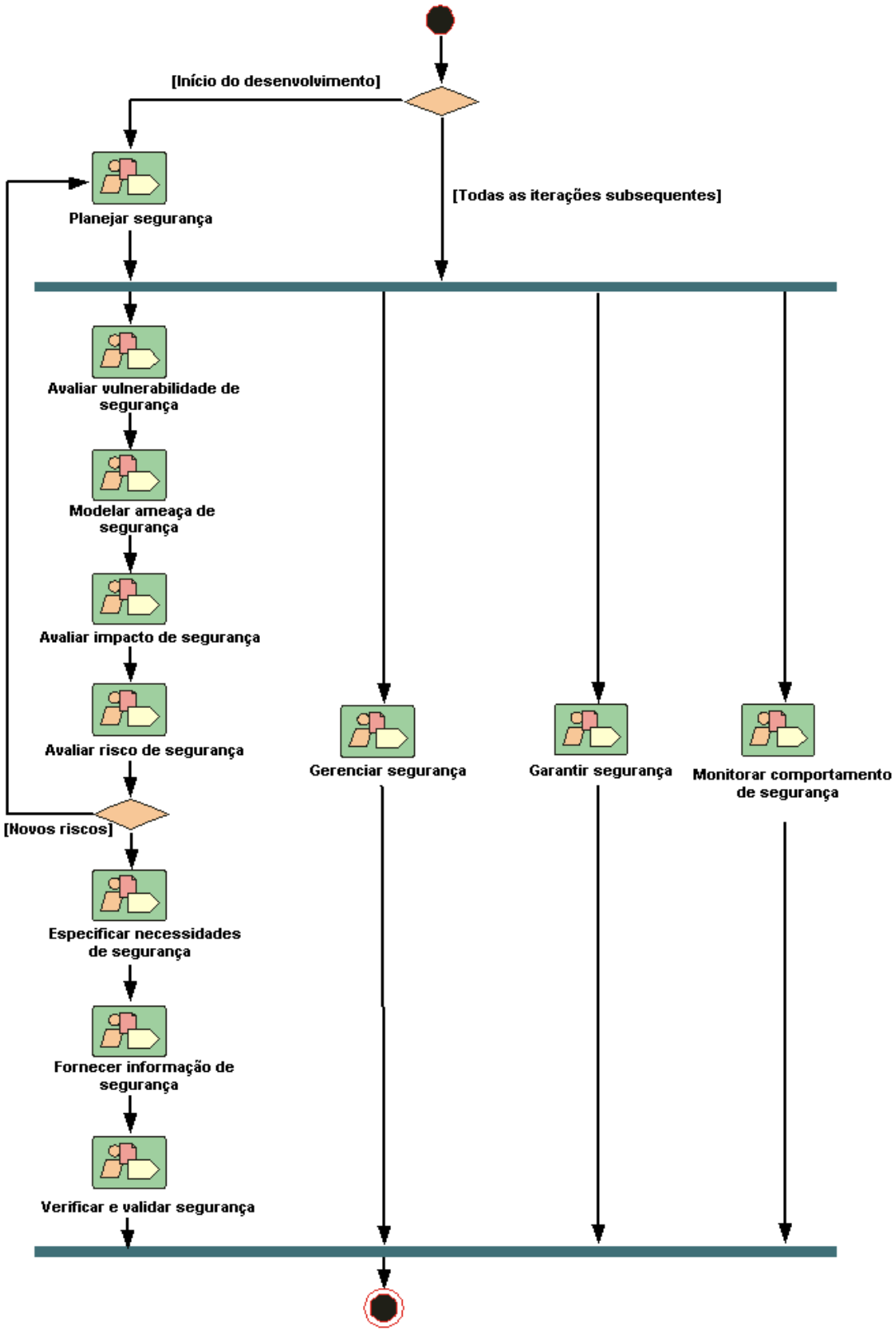

Figura 1. Processo seguro de desenvolvimento de software

3.2 Avaliar vulnerabilidade de segurança

O propósito dessa macroatividade é identificar e caracterizar (na iteração) vulnerabilidades de segurança do software relacionadas com o ambiente definido.

É intrínseco dessa macroatividade definir vulnerabilidades específicas para artefatos críticos gerados ao longo do processo de desenvolvimento, fornecendo uma avaliação geral de vulnerabilidade do software. Pela execução de métodos de identificação de vulnerabilidades, os atores envolvidos no processo obtêm uma lista detalhada de vulnerabilidades, que serão em seguida avaliadas.

Essa macroatividade é composta pelas seguintes atividades: (*i*) Identificar vulnerabilidades de segurança; e (*ii*) Analisar as vulnerabilidades de segurança identificadas.

3.3 Modelar ameaça de segurança

O propósito dessa macroatividade é identificar e caracterizar ameaças de segurança com suas propriedades e características, a partir das vulnerabilidades resultantes da macroatividade anterior.

Uma ameaça pode ser definida como um evento que possa comprometer o funcionamento habitual do software, impactando em seus objetivos em vários níveis. 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. Isto permite o desenvolvimento de estratégias de redução dessas ameaças, as quais são implementadas em projeto, código, e casos de teste.

Essa macroatividade é composta 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.

3.4 Avaliar impacto de segurança

O propósito dessa macroatividade é identificar e caracterizar impactos que sejam relevantes com respeito ao software e avaliar a possibilidade da ocorrência desses impactos.

Os impactos de segurança podem ser tangíveis, como prejuízos financeiros, ou intangíveis, como perda de reputação. Entenda-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.

Essa macroatividade é composta pelas seguintes atividades: (*i*) Tratar atividades críticas para segurança; (*ii*) Revisar artefatos do software que impactam na segurança; e (*iii*) Identificar e descrever impactos de segurança.

3.5 Avaliar risco de segurança

O propósito dessa macroatividade é, de posse das informações levantadas de vulnerabilidade, ameaça e impacto da iteração, avaliar os riscos inerentes do software em desenvolvimento, pela identificação da exposição de segurança, o risco dessa exposição, e a priorização desses riscos. A avaliação de risco atua analisando as exposições, que podem impedir o protótipo do software de alcançar seus objetivos.

Essa macroatividade é composta pelas seguintes atividades: (*i*) Identificar exposição de segurança; (*ii*) Avaliar risco de exposição de segurança; e (*iii*) Priorizar riscos de segurança.

3.6 Especificar necessidades de segurança

O propósito dessa macroatividade é especificar as necessidades relacionadas com segurança para o software entre todos os envolvidos (principalmente o usuário), em cada iteração.

Essa macroatividade é composta 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.

3.7 Fornecer informação de segurança

O propósito dessa macroatividade é prover aos arquitetos de software, projetistas, implementadores, ou usuários informações adicionais de segurança de que eles necessitam.

Geram-se informações de segurança para que:

- o máximo de problemas do software sejam revisados e resolvidos em relação a implicações de segurança;
- os membros da equipe de projeto compreendam o processo seguro e entendam aspectos da segurança da informação, para que possam realizar suas funções; e
- possíveis soluções dadas a problemas de segurança, sejam baseadas em informações pertinentes.

Essa macroatividade é composta pelas seguintes atividades: (*i*) Entender necessidades de informação de segurança; (*ii*) Identificar restrições e ponderações de segurança; (*iii*) Fornecer requisitos de apoio ao processo seguro; e (*iv*) Fornecer alternativas de segurança.

3.8 Verificar e validar segurança

O propósito dessa macroatividade é garantir que as soluções sejam verificadas e validadas em relação à segurança ao longo do processo de desenvolvimento.

Procura-se garantir que soluções sejam verificadas em relação a requisitos de segurança, arquitetura e projeto, usando-se observação, demonstração, análise e teste. Também é necessário que essas soluções 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 às necessidades de segurança do cliente.

Essa macroatividade é composta 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.

3.9 Gerenciar segurança

O propósito dessa macroatividade 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 os quais se integram com os controles existentes.

Define-se como será o gerenciamento da segurança, envolvendo a definição de treinamentos e programas de educação de segurança necessários. Os serviços de segurança e mecanismos de controle para o projeto de desenvolvimento específico são detalhados.

Essa macroatividade é composta 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.

3.10 Garantir segurança

O propósito dessa macroatividade é definir um conjunto de atividades, que possam ser aplicadas para garantir que a segurança do produto seja alcançada e mantida.

Os envolvidos com o projeto deveriam receber informações que assegurem que suas expectativas em relação ao uso adequado do processo seguro e à segurança do software em desenvolvimento sejam atingidas. Por isso, é necessário assegurar que controles eficazes sejam definidos e implementados, para garantir que artefatos críticos sejam devidamente protegidos.

Essa macroatividade é composta pelas seguintes atividades: (*i*) Definir estratégia de garantia de segurança; (*ii*) Conduzir análise de impacto das mudanças na segurança; e (*iii*) Controlar as evidências da garantia de segurança.

3.11 Monitorar comportamento de segurança

O propósito dessa macroatividade 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 "brecha" de segurança sejam monitorados e, por conseguinte, identificados, reportados e acompanhados até suas correções.

Os ambientes externo e interno devem ser monitorados em relação a fatores que possam ter impactos na segurança do software. Portanto, objetiva-se que:

- eventos externos e internos relacionados com segurança sejam detectados e acompanhados;
- incidentes sejam tratados de acordo com a política; e
- mudanças na postura de segurança operacional sejam identificadas e manipuladas de acordo com os objetivos de segurança.

Essa macroatividade é composta 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.

Na próxima seção, será detalhada a macroatividade "Especificar Necessidades de Segurança", como um exemplo mais amiúde das macroatividades apresentadas neste processo.

3.12 Macroatividade: Especificar Necessidades de Segurança

Essa macroatividade (Figura 2) define bases para a segurança do software de modo a satisfazer requisitos legais e organizacionais para segurança. Essas necessidades baseiam-se no contexto de segurança operacional do software, nos objetivos de segurança, no atual ambiente de segurança e no sistema da organização.

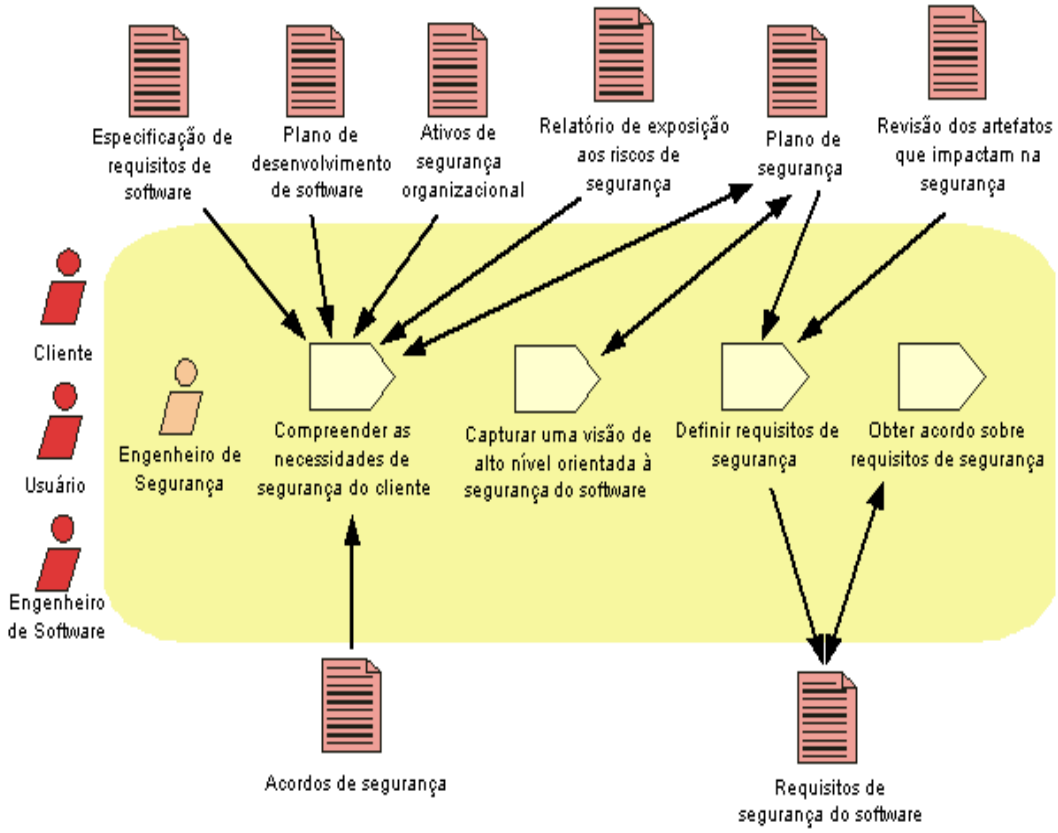

Figura 2. Macroatividade: Especificar necessidades de segurança

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. Essa visão auxilia na definição de requisitos de segurança. O *Engenheiro de software* realiza medições para obter acordos sobre esses requisitos de segurança. Cada uma de suas quatro atividades é detalhada abaixo.

3.12.1 Compreender as necessidades de segurança do cliente

O objetivo dessa atividade é coletar as informações necessárias para um entendimento das necessidades de segurança do cliente para o software em desenvolvimento. Essas necessidades são influenciadas segundo sua importância para o cliente, do risco de segurança, e 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 2000]. Convém que controles e responsabilidades específicos para atender a essas necessidades sejam definidos e documentados.

Essa atividade possui os seguintes passos: (*i*) Identificar leis, políticas, padrões, restrições, e influências externas que se relacionam com o software; (*ii*) Restringir acesso à informação; e (*iii*) Identificar o propósito do software para determinar seu contexto de segurança.

3.12.2 Capturar uma visão de alto nível orientada à segurança do software

O objetivo desta atividade é desenvolver uma visão de alto nível orientada à segurança do software, incluindo papéis, responsabilidades, ativos, recursos, proteção pessoal, proteção física, e operação.

Essa visão 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 ambiente.

Essa atividade possui os seguintes passos: (*i*) Levantar premissas; e (*ii*) Capturar objetivos de alto nível que definem a segurança do software.

3.12.3 Definir requisitos de segurança

O objetivo desta atividade é definir um conjunto consistente de requisitos que estabelecem o nível de segurança a ser implementado no software. 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 políticas, leis, padrões, necessidades de segurança, e restrições levantados para o software. A definição de requisitos deve abordar aspectos, que permitam ao produto de software detectar falhas ou comportamentos não esperados e gerenciá-los, fornecendo uma base para avaliar sua segurança 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; impactos do novo software ou do software modificado na segurança da informação da empresa; desenvolvimento ou manutenção do novo software existente não afeta de forma adversa sistemas existentes; e o software tem aderência a leis e normas da empresa.

3.12.4 Obter acordo sobre requisitos de segurança

O objetivo desta atividade é obter acordo entre os envolvidos no projeto com requisitos de segurança. Cliente e usuário devem se convencer da segurança implantada através da satisfação dos requisitos de segurança. Para isso, devem ser realizadas revisões para identificar não conformidades em relação à especificação de requisitos de segurança.

Os requisitos de segurança especificados devem aplicar de forma completa e consistente as políticas de segurança, considerando as necessidades do cliente. As

causas de problemas devem ser identificadas e trabalhadas sua solução até que se chegue a um acordo.

Na seção seguinte, serão apresentados os resultados da aplicação do PSDS.

4. Aplicação do Processo Seguro

O processo seguro de desenvolvimento de software proposto foi especializado para uma organização da administração pública no Ceará, e aplicado no desenvolvimento do *Sistema de auditoria e segurança*, que foi utilizado como projeto piloto.

Este sistema trata da segurança dos acessos de todos os usuários dos sistemas corporativos dessa organização, controlando o acesso através do perfil estabelecido para cada um desses usuários. Esse sistema, que possui arquitetura *Web*, estabelece padrões para a criação, alteração e manutenção de senhas de forma segura e de acordo com a política de segurança adotada pela organização, permitindo também a configuração de dados para posterior auditoria.

Inicialmente, foram avaliadas as necessidades de segurança da informação da organização, baseando-se inicialmente nas necessidades do projeto piloto. A partir destas informações o PSDS foi especializado para a organização. Neste contexto, apenas um subconjunto de macroatividades e atividades foi mapeado.

O PSDS especializado foi apresentado à equipe do projeto piloto. Foi avaliada conjuntamente a aplicabilidade e o impacto das etapas do processo seguro nas atividades da metodologia de desenvolvimento de software da organização.

Em seguida, foi realizada uma apresentação do PSDS especializado para o patrocinador, os gestores das áreas de negócio, e o gestor de informática. O objetivo dessa apresentação foi relatar os benefícios e a importância de se aplicar o processo seguro de desenvolvimento, perceber a receptividade da alta administração, e avaliar a aplicabilidade do processo à realidade do projeto piloto.

Foram identificadas algumas peculiaridades importantes na especialização do processo seguro proposto na organização:

- Algumas das atividades propostas no processo seguro precisaram ser ajustadas, para se obter um resultado satisfatório de sua aplicação na organização.
- É importante a existência de uma prática direcionada para a identificação dos principais ativos de informação do software a ser desenvolvido.
- Não havia uma base de conhecimento (base histórica) de erros e problemas nos sistemas corporativos da organização, que permitisse identificar com maior brevidade algumas possíveis vulnerabilidades e ameaças de segurança.
- A inexperiência da equipe em conduzir a avaliação das vulnerabilidades e a modelagem das ameaças exige que se disponha de maior tempo na realização dessas atividades devido a sua importância para a identificação dos requisitos de segurança.
- A implantação do processo identificou diversas necessidades de treinamento para a equipe, incluindo um curso de práticas de programação segura.

Em relação às necessidades de segurança, seu entendimento ocorreu com o apoio do *analista de sistema* e do *engenheiro de segurança* (papel criado na

organização). Por meio da realização das atividades do processo seguro, foram identificadas as seguintes necessidades de segurança para os sistemas da organização:

- A criação automática e manual de senhas de acesso aos sistemas deve obedecer a normas internas de criação de senhas.
- Deve ser garantido que o acesso aos sistemas da organização ocorra de forma segura e que as informações dos usuários não sejam suprimidas.
- Sob hipótese nenhuma os sistemas da organização devem permitir que usuários cadastrados, inclusive aqueles com status de 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 identificados no projeto piloto foram organizados segundo as necessidades de segurança e com apoio de árvores de ataque (Figura 3) sendo apresentados no Quadro 1.

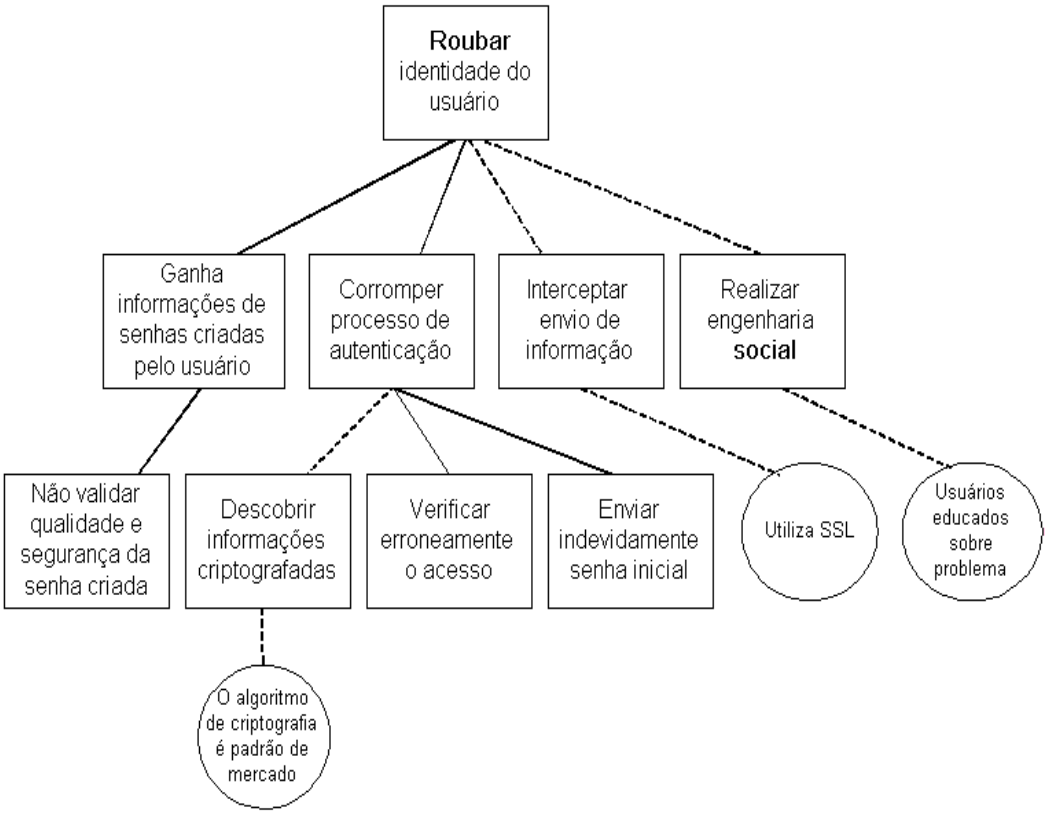

Figura 3. Exemplo de árvore de ataque do sistema de auditoria e segurança

Após a atividade "Obter acordos sobre requisitos de segurança" relativa aos requisitos de segurança identificados, foram realizadas verificações e validações de segurança. Contudo, apenas um número pequeno de testes de segurança foi conduzido em obediência às restrições de prazo para o projeto. Por exemplo, não foram conduzidos os testes de invasão. Após realização das verificações e validações de segurança, a implantação do PSDS foi dada por concluída.

Apesar da implantação com sucesso do processo seguro na organização (PSDS especializado) e da boa receptividade por parte dos gestores, ocorreram alguns problemas na condução desse processo:

- Dificuldades de entendimento e assimilação da aplicação do processo seguro. Isto demandou um maior tempo para a execução do projeto piloto.
- Obstáculos para organizar e alinhar as atividades do processo seguro com as atividades da metodologia de desenvolvimento de sistemas da organização.
- Necessidade de recursos adicionais para o processo seguro e dispositivos de integração, como forma de garantir a eficiência de sua aplicação.

Está prevista a aplicação do PSDS em uma organização que possui em torno de 350 profissionais envolvidos na área de informática, a um conjunto de sistemas considerados críticos para essa organização.

Quadro 1. Requisitos de Segurança (RS) para o Sistema de auditoria e segurança

CÓDIGO	DESCRIÇÃO
RS.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 criação de senhas. Sempre e quando o usuário compuser 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 esta respeitar as regras impostas.
RS.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.
RS.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.

5. Conclusão

A segurança de software vem se tornando um fator determinante nos sistema de informação, especialmente para aplicações Web. Para que esses sistemas sejam seguros é necessário, sobretudo, que se tenham princípios de segurança da informação, alinhados ao ciclo de vida desse software. Logo, a segurança de um aplicativo deve ser projetada e incorporada às soluções desde o início de seu desenvolvimento [Howard 2002].

O Processo Seguro de Desenvolvimento de Software (PSDS) proposto foi organizado objetivando incorporar atividades de segurança da informação no processo de desenvolvimento, a fim de apoiar a construção de sistemas mais seguros.

O PSDS está estruturado de modo a atender, principalmente, aos seguintes objetivos: (1) Prover mais visibilidade para a segurança da informação em um processo de desenvolvimento de software; (2) Tratar sistematicamente vulnerabilidades,

ameaças, impactos e riscos relacionados ao produto de software; e (3) Possibilitar que ações de segurança estejam alinhadas com os objetivos estratégicos especificados para a organização.

No entanto, a utilização do PSDS pode implicar inicialmente na adição de mais recursos e investimentos, que podem variar em conformidade com o projeto. Em contrapartida, essa adição de recursos pode resultar em produtos mais confiáveis e seguros, que busquem garantir integridade, disponibilidade e confidencialidade das informações e, por conseguinte, clientes mais satisfeitos.

Referências

Alberts, C. *et al*. (2001) "*OCTAVE - The Operationally Critical Threat, Asset, and Vulnerability Evaluation*", Carnegie Mellon – Software Engineering Institute, www.cert.org/octave.

CERT. (2005), *Coordination Center Statistics*. http://www.cert.org/stats/cert_stats.html. Acesso em julho de 2006.

CLASP - Comprehensive, Lightweight Application Security Process. (2006) Versão 1.2. Disponível em: <www.securesoftware.com/process/clasp>.

Common Criteria (2005), Version 2.3, August 2005. Disponível em: <http://www.commoncriteriaportal.org>.

Howard, M.; LeBlanc D. (2002), Writing Secure Code, 2nd edition. Microsoft Press.

ISO/IEC 15408-1. (2005a) *Information technology – Security techniques – Evaluation criteria for IT security – Part 1: Introduction and general model.*

ISO/IEC 15408-2. (2005b) *Information technology – Security techniques – Evaluation criteria for IT security – Part 2: Security functional requirements.*

ISO/IEC 15408-3. (2005c) *Information technology – Security techniques – Evaluation criteria for IT security – Part 3: Security assurance requirements.*

ISO/IEC 21827. (2002) *Information technology - Systems Security Engineering - Capability Maturity Model*.

ISO/IEC 17799. (2005) *Tecnologia da informação – Técnicas de segurança - Código de prática para a gestão da segurança da informação, ABNT, Rio de Janeiro.*

ISSEA. (2003), *International Systems Security Engineering Association*. www.issea.org. Acesso em agosto de 2003.

McGraw, Gary (2004), *Software Security*, IEEE Security and Privacy, March/April 2004, páginas 32-35.

McGraw, G.; Viega, J., (2000), "Software Security Principles: Part 4. Keep it simple, keep it private". Disponível em: <http://www-128.ibm.com/developerworks/security/library/s-simp.html>. Acesso em maio de 2004.

Paulk *et al*. 1993 Paulk, *Capability Maturity Model for Software*. Version 1.1 (CMU/SEI-93-TR-024, ADA 263403). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1993.

SSE-CMM. (2003) *System Security Engineering – Capability Maturity Model*, Version 3, www.sse-cmm.org.