Revista Eletrônica
Estácio Recife

A construção de *software* seguro: uma análise a partir dos modelos de desenvolvimento e maturidade

João Carlos D. Almeida[1], Jorge S. Correia-Neto[2], Leandro M. Queiros[3]

[1]Faculdade Frassineti – FAFIRE
Av. Cde da Boa Vista, 921 – 50.060-002 – Recife–PE – Brazil

[2]Universidade Federal Rural de Pernambuco – UFRPE - UAEADTec
Av. Dom Manoel de Medeiros, s/n, 52.171-900, Recife-PE - Brasil.

[3]Instituto Federal de Pernambuco - IFPE
Av. Prof. Luis Freire, 500, 50740-540 – Recife-PE – Brasil

jcda@cin.ufpe.br, {jorgecorreianeto, leandromarquesrr}@gmail.com

Abstract. *Security incidents have been a major concern to users and developers of information systems. It is in this context that the Building Security In Maturity Model (BSIMM) presents itself as a promising alternative, with its framework involving the following domains: governance practices, intelligence (security architecture), development and deployment. As a result, these practices lead to the accomplishment of a set of activities that conform three levels of maturity, referring to the development of secure software, established over time.*

Resumo. *Os incidentes de segurança têm sido uma preocupação constante tanto para quem utiliza como para quem desenvolve sistemas de informação. É neste contexto que o modelo de maturidade para desenvolvimento de software seguro (BSIMM) se apresenta como uma alternativa promissora, com seu framework envolvendo os seguintes domínios: práticas de governança, de inteligência (arquitetura de segurança), de desenvolvimento e de implantação. Como decorrência, estas práticas levam à realização de um conjunto de atividades que conformam três níveis de maturidade, referente ao desenvolvimento de software seguro, estabelecidos ao longo do tempo.*

1. Introdução

Num cenário no qual os sistemas de informação estão cada vez mais trocando informações entre si, principalmente via Internet, é preciso estabelecer garantias de segurança durante este processo, especialmente por que a informação é parte central do negócio das organizações e vital para a tomada de decisões (SOBRAL; PECI, 2008), sendo assim um ativo estratégico (ABNT NBR ISO/IEC 27002, 2013).

Neste sentido, a segurança das informações deve ser parte integrante da política de informações das empresas. Ou seja, como recomenda a ISO 27002 a integridade, a

confidencialidade e a disponibilidade das informações e dos dados devem ser protegidas e conduzidas por processos que garantam a segurança das informações (ABNT NBR ISO/IEC 27002, 2013). Por conseguinte, esta segurança deve ser pensada desde o momento no qual se planeja o *software*, ou um sistema computacional, até o momento no qual ele é entregue ao cliente/usuário (VIEGA; MCGRAW, 2011), já que "a segurança da informação é alcançada pela implementação de um conjunto adequado de controles, incluindo políticas, processos, procedimentos, estrutura organizacional e funções de *software* e hardware" (ABNT NBR ISO/IEC 27002, 2013, p. 4).

Neste contexto, e tendo em vista que cerca de 92% das vulnerabilidades de segurança estão no *software* (CABRAL; CAPRINO, 2015), esta pesquisa visa abordar os aspectos do desenvolvimento de *software*, sistemas e ferramentas de tecnologia da informação e comunicação (TIC) sob a ótica dos princípios e procedimentos de segurança. Neste sentido, se propõe que projetos de desenvolvimento de *softwares* sejam avaliados em relação a tais princípios.

Para tanto serão analisados e discutidos alguns dos modelos e padrões de desenvolvimento de *software* seguro mais utilizados, a partir do que apontam os trabalhos de De Win *et al.* (2008) e McGraw, Migues e West (2013). Serão analisados os processos de desenvolvimento de *software* seguro Security Development Lyfe cicle (SDL) e o Comprehensive, Lightweight Application Security Process (CLASP), além do modelo de maturidade Building Security In Maturity Model (BSIMM).

O SDL foi adotado pela Microsoft como um processo para o desenvolvimento de *software* que precisa suportar ataques maliciosos. Um conjunto de atividades focadas em segurança e de entregas em cada uma das fases do processo de desenvolvimento foi definido. Como destacam De Win *et al.* (2008), nestas atividades e entregas estão incluídos os modelos de prevenção de ameaças (para a fase de design) e a utilização de ferramentas de análise de código (tanto durante o desenvolvimento como nos testes e na manutenção).

O CLASP foi elaborado pela Open Web Application Security Project (OWASP), comunidade aberta que visa produzir e divulgar informações sobre segurança de *software* para indivíduos, organizações, universidades e governos (GONDROM; RITCHIE, 2014).

O BSIMM é utilizado para que as organizações que desenvolvem TIC possam medir se estão em conformidade com as melhores práticas de desenvolvimento de *software* seguro utilizadas no mercado, como também para analisar as demais ferramentas ofertadas para manipulação de informações negociais. Mas dado o tamanho da maioria das empresas brasileiras de TIC, especialmente de *software,* ser de pequeno ou médio porte, optou-se por estudar o BSIMM, pois o mesmo pode ser utilizado por organizações de qualquer porte (McGRAW; MIGUES; WEST, 2013).

Como aponta a Associação Brasileira das Empresas de *Software* (ABES), o mercado brasileiro de empresas de *software* é caracterizado por ter em sua maioria empresas de médio (93%) e pequeno porte, ao se analisar o número de funcionários

(ABES, 2015). Ademais, a ABES relata que existem cerca de 12.660 empresas dedicadas ao desenvolvimento, produção, distribuição de *software* e de prestação de serviço no mercado nacional, onde mais da metade destas empresas têm como atividade principal desenvolvimento e produção de *software* ou prestação de serviços. Semelhantemente, a Associação Brasileira das Empresas de Tecnologia da Informação e Comunicação (Brasscom) afirma que, em 2013, o segmento de *software* no Brasil cresceu seu valor em R$ 20,9 milhões, obtendo um crescimento de 13% em relação a 2012.

A partir deste cenário, este ensaio tem como objetivo investigar os conceitos que envolvem a construção e manutenção de *software* seguro, no contexto das micro e pequenas empresas (MPE) que desenvolvem ou usam *software*. Como ainda não existe um consenso acerca da utilização de um modelo de desenvolvimento de *software* seguro ou de maturidade no desenvolvimento de *software* seguro no contexto das MPE, justifica-se um aprofundamento maior na análise das vantagens e desvantagens dos modelos existentes, sendo esse o problema central de pesquisa.

Para viabilizar esta análise, os autores apresentam o contexto do estudo (as diversas formas usuais de segurança da informação), os modelos de referência e, por fim, as considerações finais.

2. Contexto do estudo

As tecnologias da informação e da comunicação (TIC), e a Internet em especial, têm mudado o papel que o *software* exerce nos negócios, pois tanto uma falha operacional como "uma falha de segurança afetam diretamente os objetivos de negócio" (LEIVA, 2014, online). Isso demonstra o importante papel da TI tanto para pequenas como para médias e grandes organizações em suas rotinas administrativas (IBGE, 2010), notadamente aquelas que realizam *e-business* (VIEGA; MCGRAW, 2011).

Pesquisa realizada com micro e pequenas empresas (MPE) sobre o uso das TIC verificou que, das MPE que utilizam o computador, a maioria utilizava *softwares* prontos, adquiridos ou alugados de terceiros (96,6%) (IBGE, 2010). Como principais funcionalidades apontadas nessa pesquisa têm-se "a gestão eletrônica de documentos, 85,6%, e a gestão comercial, 63,9%, uma vez que atividades de administração e vendas são atividades presentes em quaisquer tipos de empresas" (IBGE, 2010, p. 38).

Contudo, estes sistemas podem não estar seguindo as recomendações da ISO 27002, que aponta que a segurança oferecida somente por meios técnicos é limitada e necessita ser apoiada por gerenciamento e procedimentos adequados. Pesquisa sobre segurança na América Latina em 2014, com mais de 3980 executivos, revela que suas empresas, ao adotarem medidas de proteção contra incidentes de segurança, têm como prática comum a adoção de antivírus, *firewall* ou *antispam* (ESET, 2014). Tal prática foi observada entre 57% a 82% das empresas que tiveram dados coletados. Além disso, o método de autenticação de sistemas e redes de detecção de intrusão teve uso

substancialmente menor quando comparada com a ótica de incidentes que envolve seus empregados.

Nessa mesma linha, o Grupo de Resposta a Incidentes de Segurança para a Internet Brasileira (CERT.br) destaca o forte aumento dos incidentes de segurança nos últimos anos, sendo o Brasil o país que teve o maior percentual (75,38%) de ataques (CERT.br, 2015), o que reforça a necessidade de se investir nesse contexto.

Por esta razão, para se ofertar um melhor gerenciamento das informações necessárias à continuidade do negócio, é preciso mapear os ativos da informação (COBIT-5, 2012). Este tipo de mapeamento é um processo interativo que evolui com o tempo, e que de acordo com guia brasileiro de referência para a segurança das infraestruturas críticas da informação, do Governo Federal (GUIA SICI, 2010), envolve a identificação e a classificação de ativos de informação, a identificação de potenciais ameaças e vulnerabilidades e a avaliação de riscos.

Desta maneira, a organização terá embasamento para estabelecer ações e controles na segurança das informações e este mapeamento poderá servir de *baseline* para o planejamento das ações necessárias no estabelecimento da segurança da informação nos processos organizacionais. Afinal, segurança não é algo a ser adquirido de forma acabada, pois envolve políticas, pessoas, processos e tecnologia (SILVA; ARAÚJO; AZEVEDO, 2014).

Portanto, no que se refere ao desenvolvimento de *software* as empresas deverão observar as métricas de *software* sob o ponto de vista da segurança da informação visando promover a sua proteção, já que segundo o National Institute of Standards and Technology (NIST), 92% das vulnerabilidades estão no *software* (CABRAL; CAPRINO, 2015).

Diante disto, para que as empresas que desenvolvem *software* construam seus produtos pensando em critérios de segurança da informação, será necessário reavaliar tanto o processo de desenvolvimento como também o nível de maturidade da organização, o que inclui uma análise do ambiente interno e estabelecimento de um modelo organizacional. Neste sentido, haverá a necessidade de avaliação, gerenciamento de riscos e estabelecimento de controle da TIC organizacional por meio de uma governança corporativa da TIC, que pode ter como referência o modelo de maturidade oferecido no COBIT. Entretanto, a organização que almeja amadurecer nesta área, sob o a ótica da segurança da informação, precisará buscar outros modelos de maturidade para incrementar e servir de referência a fim de alcançar este objetivo, já que a governança é uma ação importante e será um dos pilares das ações direcionadas para desenvolver códigos seguros (OPENSAMM, 2008).

Além disso, o problema da segurança de *software* é um problema de gestão de riscos que, para ser solucionado, é preciso começar cedo, mapear as ameaças e arquitetar a segurança (VIEGA; MCGRAW, 2011). A seguir são apresentados os principais modelos identificados na literatura atual.

3. Modelos de referência

Neste ensaio teórico serão feitas reflexões sobre os modelos SDL e CLASP para o desenvolvimento de *software* seguro e sobre o modelo de maturidade BSIMM, nas subseções a seguir.

3.1. O modelo SDL

Como parte do direcionamento traçado pela Microsoft em 2002, acerca da segurança em seus produtos, o SDL passou a ser implementado consistentemente. O conjunto de atividades que o compõe pode ser assim resumido (DE WIN et al., 2008):

- Segurança como uma qualidade suportiva: o objetivo principal do SDL é aumentar a qualidade do *software* direcionado a funcionalidade aumentando sua postura de segurança. O SDL é um *add-on* ao processo de desenvolvimento de *software*;
- Processo bem definido: o processo de SDL é bem definido e suas atividades são agrupadas em estágios. Apesar de serem específicas de segurança, estão mapeadas nas fases de desenvolvimento do *software.* Além disso, várias atividades do SDL se dão de forma contínua, como modelagem de ameaças e educação;
- Boa orientação: o SDL define claramente o método a ser utilizado para realizar as atividades, de forma bem pragmática, facilitando o trabalho até dos menos experientes;
- Perspectiva de gestão: o SDL toma uma perspectiva de gestão para a elicitação e a descrição das atividades. Isso é ótimo, tendo em vista a complexidade inerente às questões de segurança, e mostra que segurança como uma qualidade precisa ser gerenciada para ser realizada na prática.

Além disso, concluem os citados autores, antes que o *software* sujeito ao SDL possa ser liberado, ele deve ser submetido a uma revisão final de segurança por uma equipe independente do seu grupo de desenvolvimento. Quando comparado ao *software* que não foi sujeito ao SDL, o que passou pelo SDL experimentou uma taxa significativamente reduzida de descoberta externa de vulnerabilidades de segurança.

Por fim, vale salientar que mesmo que o método venha a ser customizado para necessidades específicas, o National Institute of Standards and Technology (NIST) recomenda que as organizações incorporem estes aspectos de segurança das TIC em seus processos de desenvolvimento (GRANCE; HASH; STEVENS, 2006).

3.2. O modelo CLASP

O modelo Comprehensive, Lightweight Application Security Process (CLASP) indica 7 melhores práticas em segurança de aplicações, práticas estas que cobrem todo o processo de desenvolvimento e uso de *software* de maturidade ajudam a organização a

melhorar sua destreza em certa área de atuação (VIEIRA NETO, 2011). Seguindo o que apontam De Win *et al.* (2008), o modelo CLASP tem as seguintes características:

- Segurança como ponto focal: a CLASP suporta o desenvolvimento de *software* onde se deseje a segurança como tomando um papel central. Abrange atividades com uma perspectiva teórica de segurança e seguem por atividades de amplo expectro;
- Estrutura limitada: são atividades independentes, mas que devem se integrar no processo de desenvolvimento e no ambiente de operação. Dois *road maps* (legado e "campo verde") foram definidos para guiar a combinação de atividades em um conjunto ordenado e coerente;
- Baseado em papéis: devem ser definidas funções que podem ter impacto na postura de segurança do produto e atribuidas atividades a essas funções. Os papéis são responsáveis pela finalização e qualidade dos resultados de uma atividade. Como tal, os papéis são usados como uma perspectiva adicional para estruturar o conjunto de atividades;
- Rico em recursos: seu conjunto de recursos de segurança facilita e apoia a implementação das atividades, a exemplo de sua lista de vulnerabilidades, que pode ser usada durante as revisões de código.

O modelo CLASP é implementado em camadas ou visões, como explica Vieira Neto (2011):

- Conceitual: apresenta uma visão geral do processo e de como seus componentes interagem. São introduzidas as melhores práticas, sua interação com as políticas de segurança, alguns conceitos de segurança e os componentes do processo;
- Regras: a visão baseada em regras define as responsabilidades básicas de cada membro do projeto relacionando-os com as atividades propostas, assim como a especificação de quais são os requisitos básicos de cada função;
- Avaliação de Atividades: descreve o propósito de cada atividade, o custo de implementação, a aplicabilidade, o impacto relativo aos riscos em caso de não se aplicar a atividade;
- Implementação: descreve o conteúdo das 24 atividades de segurança e identifica os responsáveis por sua implementação, bem como as atividades relacionadas;
- Vulnerabilidades: seu catálogo de 104 tipos de vulnerabilidades no desenvolvimento de *software* envolve diversos tipos de erros, mas também são utilizadas técnicas de mitigação e avaliação de risco.

3.3. Os modelos de maturidade e o modelo BSIMM

Os modelos de maturidade ajudam a organização a melhorar sua destreza em certa área de atuação. No tocante à maturidade da gestão da informação os modelos de

maturidade oferecem estágios/pilares para a evolução no amadurecimento dos processos de desenvolvimento seguro de *software*.

Os modelos de maturidade para desenvolvimento de *software* seguro sugerem uma estrutura organizacional com pelo menos um pilar para a governança e os demais para arquitetura, *compliance* (conformidade) de ações e requisitos de segurança. Por exemplo, no Software Assurance Maturity Model (SAMM) existem os seguintes pilares: Requirements & Design, Verification & Assessment e Deployment & Operations. Semelhantemente, em outro modelo também de licença aberta, o Building Security In Maturity Model (BSIMM-V), são estabelecidos quatros domínios: Governance, Intelligence, SSDL Touchpoints e Deployment (MCGRAW; MIGUES; WEST, 2013).

Estes modelos deixam claro que mudanças estruturais nas organizações favorecem o processo de desenvolvimento de *software*s seguros. Desta forma, o modelo de maturidade para desenvolver *software* seguro que será objeto de aprofundamento neste artigo será o BSIMM-V por ser um *framework* mais dinâmico e aderente à realidade das empresas, pois é resultante da interação com o mercado através da coleta de dados no ambiente das empresas ao longo do tempo (MCGRAW; MIGUES; WEST, 2013). Logo, a empresa poderá ter um direcionamento que estabelecerá diretrizes e estratégias no tratamento de risco e na solução dos problemas de segurança, tendo em vista que o maior problema de segurança num computador são os profissionais que não sabem identificar qual é o problema e afirmam ser somente o *software* (VEIGA; MCGRAW, 2011).

Além disso, a segurança não deve ser algo que é adicionado num *software* que já foi construído, pois ela se caracteriza como uma propriedade comportamental de um sistema num ambiente particular (VIEGA; MCGRAW, 2011). Na mesma linha, pode-se afirmar que é melhor arquitetar a segurança como um projeto inicial do que tentar adicioná-la a um projeto já em execução, pois o ambiente no qual o *software* será inserido é um mundo particular no qual qualquer mudança pode gerar problemas diversos (VIEGA; MCGRAW, 2011). O BSIMM é um modelo que vem evoluindo desde 2008 e que atende às práticas do processo de desenvolvimento de *software* seguro existentes nas organizações (MCGRAW; MIGUES; WEST, 2013; ROMANIZ et al., 2013). Como este modelo exige empenho (FREIRE, 2014) acaba auxiliando na análise da situação atual e futura, explicitando os esforços a serem ainda realizados (COLOMBO, 2013).

O BSIMM é classificado como um modelo de maturidade, pois ajuda as organizações a aperfeiçoarem a segurança de *software* ao longo do tempo (MCGRAW; MIGUES; WEST, 2013). Além disso, sua quinta versão inclui descrições de atividades atualizadas, uma nova atividade e dados de 67 empresas, além de um estudo longitudinal (FREIRE, 2014).

Entretanto, para manter alinhadas as iniciativas de segurança o modelo faz uso de um *framework* que permite descrever de maneira uniforme essas iniciativas (MCGRAW; MIGUES; WEST, 2013). Ele é denominado de *framework* de segurança

de *software* (SSF, sigla em inglês) apresentado a seguir. Assim sendo, o BSIMM apresenta 112 atividades que podem ser utilizadas por qualquer organização. Estas atividades são descritas nos termos do SSF, que identifica práticas agrupadas dentro de domínios estabelecidos. Desta maneira, as organizações poderão se avaliarem e verificar em que nível se encontram dentro de cada prática e onde querem chegar.

Entretanto, o modelo deixa claro que para o sucesso de iniciativas de segurança é necessário, além do patrocínio da alta administração, ter um grupo de segurança de *software* (SSG) (MCGRAW; MIGUES; WEST, 2013). Desta forma, as realizações das atividades descritas no *framework* devem estar sob a condução do SSG que deve existir antes de qualquer planejamento e execução de atividades descritas no BSIMM. Contudo, se não houver um SSG recomenda-se criá-lo, ficando a cargo da empresa o quantitativo de membros deste grupo. Porém, é de suma importância que neste grupo estejam pessoas que conheçam de desenvolvimento de *software* e que possuam alguma experiência nesta área (MCGRAW; MIGUES; WEST, 2013).

3.3.1 *Framework* de segurança de *software*

O *framework* de segurança de *software* (SSF), com suas doze práticas agrupadas em quatro domínios, serve de base para que os objetivos do BSIMM sejam alcançados. Cada domínio do SSF pode ser entendido como uma área de atuação, definida pela organização, em sua estrutura de negócio. Desta forma, existem três práticas definidas para cada domínio, como apresenta a figura 1.

Governança	Inteligência	SSDL Touchpoints	Implantação
Estratégia e Métricas	Modelos de Ataque	Análise Arquitetural	Testes de Penetração
Conformidade e Política	Recursos e Projeto de Segurança	Revisão de Código	Ambiente de Software
Treinamento	Padrões e Requisitos	Testes de Segurança	Gestão de Configuração e Gestão de Vulnerabilidade

Figura 1. *Framework* de segurança de *software* (SSF).

Fonte: BSIMM-V (2013).

Assim também, o BSIMM apresenta metas de negócio para cada prática que poderá auxiliar a organização na adoção de práticas que fazem sentido para ela e para sua realidade de negócio. Consoante a isto, a figura 2 contém uma descrição das metas associadas a cada prática.

Domínio	Metas de Negócio
Governança	Transparência, Responsabilização, Pesos e Contrapesos
Inteligência	Auditabilidade, Diligência, Padronização
SSDL Touchpoints	Controle da Qualidade
Implantação	Controle da Qualidade, Gestão de Mudanças

Figura 2. Metas de alto nível de cada domínio.

Fonte: BSIMM-V (2013).

Assim, a figura 3 contém uma descrição detalhada e uma visão holística da relação entre as práticas dentro de cada domínio com cada meta associada.

Por fim, cada prática está organizada em três níveis, os quais comportam as atividades que contribuem para cada nível dentro das metas de negócio. Para escolher quais atividades do BSIMM a organização adotará é sugerido criar uma estratégia juntamente com um plano de segurança de *software* que foque nas metas e objetivos como ponto de partida (MCGRAW; MIGUES; WEST, 2013). Além disso, criar uma base de conhecimento sobre segurança através de treinamento para os envolvidos no ciclo de desenvolvimento do *software* se configura como uma boa prática fundamental para se criar uma cultura de segurança na corporação.

Por esta razão, as atividades do BSIMM podem servir de *baseline* para que as organizações verifiquem se suas próprias ações estão em conformidade tanto com o mínimo comum observado nas empresas estudadas como com as demais atividades do modelo.

Domínio	Prática	Metas de Negócio
Governança	Estratégia e Métricas	Transparência das expectativas, Responsabilidade pelos resultados
	Conformidade e Política	Orientações normativas para todos os stakeholders, Responsabilização
	Treinamento	Força de trabalho bem informada, Correção de erros
Inteligência	Modelos de Ataque	Conhecimento personalizado
	Recursos e Projeto de Segurança	Padrões reutilizáveis, Orientações normativas para todos os stakeholders
	Padrões e Requisitos	Orientações normativas para todos os stakeholders
SSDL Touchpoints	Análise de Arquitetura	Controle da qualidade
	Revisão de Código	Controle da qualidade
	Testes de Segurança	Controle da qualidade
Implantação	Testes de Penetração	Controle da qualidade
	Ambiente de Software	Gestão de mudanças
	Gestão de Configuração e Gestão de Vulnerabilidade	Gestão de mudanças

Figura 3. Visão Holística das metas de alto nível de cada domínio.

Fonte: BSIMM-V (2013).

4. Considerações finais

Num ambiente no qual existem interações em proporções crescentes, sempre conectados por redes, os sistemas de informação têm sido frequentemente expostos a incidentes de segurança. Com o intuito de promover a proteção de suas informações, muitas organizações têm investido em barreiras tais como *firewall* e antivírus, entretanto, grande parte das vulnerabilidades de segurança está presente no *software*.

Este fato demanda uma mudança no eixo da ação referente à garantia da segurança da informação, ou seja, a segurança deixa de ser algo apenas implantado após a construção ou compra do *software* para ser uma preocupação existente em todo o seu processo de desenvolvimento. Consequentemente, isto demanda mudanças nas práticas de desenvolvimento de *software,* que agora precisa ser estabelecida por meio de políticas e controles que reduzam o risco de incidentes de segurança.

Por estas razões, a adoção de um modelo de maturidade de desenvolvimento de *software* seguro é um ponto fundamental para a organização que almeja amadurecer e otimizar suas iniciativas de segurança. Por isso, o BSIMM foi o modelo descrito neste trabalho como um referencial para as empresas que desenvolvem *software* e que pretendem ofertar produtos com um maior nível de qualidade no tocante à segurança da informação.

Ainda mais, o *framework* de segurança de *software* (SSF) é o produto mais importante do BSIMM, pois estrutura práticas e atividades dentro de áreas de concentração, aqui denominadas de domínios, que auxiliam na condução dos processos que contribuem para o amadurecimento da organização no sentido de desenvolver sistemas de informação seguros. Entretanto, não se pode esquecer de criar ou manter um grupo de segurança de *software* (SSG) para condução das atividades do *framework*, pois é importante ter gente preocupada e responsável pelo controle interno da política de segurança e pelo processo de desenvolvimento de *software* seguro da empresa.

Portanto, independente do modelo de desenvolvimento escolhido, SDL ou CLASP, o BSIMM é um modelo de maturidade para desenvolvimento de *software* seguro que qualquer organização, de pequeno, médio ou grande porte, pode aplicar à sua realidade de iniciativas de segurança visando à verificação da conformidade de suas práticas de desenvolvimento de *software* sobre o ponto de vista da segurança da informação.

Referências

ABNT. NBR ISO/IEC 2002. Tecnologia da Informação - Técnicas de Segurança – Código de Prática para Controles de Segurança da Informação. Rio de Janeiro: ABNT, 2013.

ABES. ASSOCIAÇÃO BRASILEIRA DE EMPRESAS DE *SOFTWARE*. Mercado Brasileiro de *Software*: panorama e tendências, 2015. São Paulo: ABES, 2015. Disponível em:

<http://central.abes*software*.com.br/Content/UploadedFiles/Arquivos/Dados%202011/ABES-Publicacao-Mercado-2015-digital.pdf>. Acesso em: 16 out. 2015.

BRASIL. Guia de referência para a segurança das infraestruturas críticas da informação (GUIA SICI) / Gabinete de Segurança Institucional, Departamento de Segurança da Informação e Comunicações. CANONGIA, Claudia; GONÇALVES JÚNIOR, Admilson; MANDARINO JUNIOR, Raphael (Orgs.) – Brasília: GSIPR/SE/DSIC, 2010.

BRASSCOM - Associação Brasileira das Empresas de Tecnologia da Informação e Comunicação. Inteligência de Mercado. Disponível em: <http://www.brasscom.org.br/brasscom/Portugues/detInstitucional.php?codArea=3&codCategoria=56>. Acesso em: 17 out. 2015.

BRASSCOM - Associação Brasileira das Empresas de Tecnologia da Informação e Comunicação. Governo Digital. Disponível em: <http://www.brasscom.org.br/brasscom/Portugues/detInstitucional.php?codArea=6&codCategoria=69>. Acesso em: 17 out. 2015.

CABRAL, Carlos; CAPRINO, Willian. Trilhas em Segurança da Informação: caminhos e ideias para a proteção de dados. Rio de Janeiro: Brasport, 2015.

CERT.BR. Estatísticas dos Incidentes Reportados ao CERT.br. Disponível em: <http://www.cert.br/stats/incidentes/>. Acesso em: 20 set. 2015.

COLOMBRO, Regina M. Thienne. Proposta de uma Metodologia de Medição e Priorização de Segurança de Acesso para Aplicações Web (Tese). 2013. In: p. 56. Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Produção, São Paulo.

DE WIN, Bart; SCANDARIATO, Riccardo; BUYENS, Koen; GRÉGOIRE, Johan; JOOSEN, Wouter. On the secure software development process: CLASP, SDL and Touchpoints compaired. Journal of Information and Software Technology, v. 51, p. 1152-1171, 2009.

ESET. Security Report Latinoamérica 2014. Disponível em: <http://www.welivesecurity.com/wp-content/uploads/2014/06/informe_esr141.pdf>. Acesso em: 15 set. 2015.

ESET. Security Report Latinoamérica 2015. Disponível em: <http://www.welivesecurity.com/wp-content/uploads/2015/01/tendencias_2015_eset_mundo_corporativo.pdf>. Acesso em: 15 set. 2015.

FREIRE, Matheus da Silva. Ferramenta de Apoio a Auditoria de Programa de Segurança de *Software*. 2014. In: p.23. UNB, Brasília. Disponível em: < http://bdm.unb.br/bitstream/10483/8765/1/2014_MatheusdaSilvaFreire.pdf>. Acesso em: 18 de Ago. 2015.

GRANCE, Tim; HASH, Joan; STEVENS, Marc. Security Considerations in the Information System Development Life Cycle, Recommendations of the National Institute of Standards and Technology, Special Publication 800-64 REV. 1, 2006.

LEIVA, Marcelo. IDC: La Inversión en Soluciones de *Software* de Seguridad en Latinoamérica Fue de 485 Millones de Dólares Durante 2014. Disponível em: <http://br.idclatin.com/releases/news.aspx?id=1866>. Acesso em: 17 out. 2015.

GONDROM, Tobias; RITCHIE, Paul. OWASP 2014 Annual Report – growing, learning, sharing, leading. Acesso em: 9 jan. 2017. Disponível em: https://www.owasp.org/images/7/7e/2014_OWASP_Annual_Report_Final.pdf

IBGE. Instituto Brasileiro de Geografia e Estatística. Pesquisa sobre o Uso das Tecnologias da Informação e Comunicação nas Empresas - 2010. Rio de Janeiro, 2012. Disponível em: <http://biblioteca.ibge.gov.br/visualizacao/livros/liv62955.pdf>. Acesso em: 17 out. 2015.

IT GOVERNANCE INSTITUTE. COBIT 4rd Edition: Control Objectives. Estados Unidos: Information Systems Audit and Control Association. 2007.

IT GOVERNANCE INSTITUTE. COBIT 5rd Edition: Control Objectives. Estados Unidos: Information Systems Audit and Control Association. 2012.

McGRAW, G.; MIGUES, S.; WEST, J. Building Security In Maturity Model, 2013. Disponível em: <https://www.bsimm.com/download/>. Acesso em: 20 jul. 2015.

OPENSAMM. *Software* Assurance Maturity Model. Disponível em: <http://www.opensamm.org/downloads/SAMM-1.0.pdf>. Acesso em: 5 jan. 2015.

REDE SEGURA. Gerenciamento de Vulnerabilidades: Estratégia e Planejamento do Processo. Disponível em: <http://www.redesegura.com.br/gerenciamento-de-vulnerabilidades/estrategia-e-planejamento-do-processo/>. Acesso em: 15 set. 2015.

REDE SEGURA. Gerenciamento de Vulnerabilidades: Modelos de Maturidade em Segurança. Disponível em: <http://www.redesegura.com.br/gerenciamento-de-vulnerabilidades/modelos-de-maturidade-em-seguranca/>. Acesso em: 15 set. 2015.

ROMANIZ, Susana; RAMOS, Juan Carlos; CASTELLARO, Marta; RAMOS, Ignacio. Catalogación como Apoyo al Uso de Patrones de Seguridad. In: 5to Workshop de Seguridad Informatica, WSegI 2013. Disponível em: < http://42jaiio.sadio.org.ar/proceedings/simposios/Trabajos/WSegI/02.pdf>. Acesso em: 17 out. 2015.

SILVA, Narjara Bárbara Xavier; ARAÚJO, Wagner Junqueira de; AZEVEDO, Patrícia Morais de. Engenharia social nas redes sociais online: um estudo de caso sobre a exposição de informações pessoais e a necessidade de estratégias de segurança da informação. Revista Ibero-Americana de Ciência da Informação, [S.l.], v. 6, n. 2, mar. 2014. ISSN 1983-5213. Disponível em:

<http://www.periodicos.unb.br/index.php/RICI/article/view/9222/7660>. Acesso em: 09 jan. 2017.

SOBRAL, Filipe; PECI, Alketa. Administração: Teoria e Pratica no Contexto Brasileiro. São Paulo: Pearson Prentice-Hall, 2008.

VIEGA, John; MCGRAW, Gary. Building Secure *Software*: How to Avoid Security Problems the Right Way. Estados Unidos: Crawfordsville, Indiana. Addison-Wesley, 2011.Holton, M. and Alexander, S. (1995) "Soft Cellular Modeling: A Technique for the Simulation of Non-rigid Materials", Computer Graphics: Developments in Virtual Environments, R. A. Earnshaw and J. A. Vince, England, Academic Press Ltd., p. 449-460.

VIEIRA NETO, Tarcizio. Modelo de processo para desenvolvimento de aplicações seguras. Acesso em: 09 jan. 2017. Disponível em: https://www.owasp.org/images/f/f7/TarcizioNeto_Apresentacao_PROSEG_AppSecLatam2011.pdf