Título Público da Pesquisa: Um estudo de caso sobre o uso de técnicas e ferramentas para detecção de vulnerabilidades em organizações que utilizam métodos ágeis.

Título Principal da Pesquisa: Um estudo de caso sobre o uso de técnicas e ferramentas para detecção de vulnerabilidades em organizações que utilizam métodos ágeis.

Pesquisador Principal: .

Desenho da Pesquisa: Atualmente, as organizações operam em um ambiente com constantes mudanças culturais, sociais, políticas e econômicas. Com isso elas devem responder rapidamente a essas mudanças e pressões competitivas (SOMMERVILLE, 2011).

Processos de desenvolvimento de software que planejam especificar completamente os requisitos e, em seguida, projetar, construir e testar o sistema de forma sequencial e rígida acabam sendo inadequados para ambientes que sofrem constantes mudanças (SOMMERVILLE, 2011).

Se os requisitos de software mudam, consequentemente o projeto, implementação e testes do sistema mudam também. Por conta disso, torna-se difícil obter um conjunto completo de requisitos estáveis de software (PRESSMAN, 2009). No entanto, em ambientes com constantes mudanças, uma abordagem ágil é recomendável.

Devido à agilidade do processo de desenvolvimento e pressão na entrega, alguns softwares não têm uma análise completa da segurança para cada fase do ciclo de vida e com isso podem surgir vulnerabilidades (OWASP, 2010). De acordo com a Owasp (2010), a maioria das vulnerabilidades nos sistemas de software são causadas pela falta de aplicação de políticas de segurança no projeto. Para evitar isso, é necessário analisar o uso de práticas de segurança pelas organizações, as habilidades dos membros e as necessidades de formação da equipe, bem como adaptar os processos de segurança de desenvolvimento de software tradicional aos processos dos métodos ágeis (HOWARD; LIPNER, 2006).

Howard e Lipner (2009) apontam que práticas de segurança incorporadas no ciclo de vida de desenvolvimento de software diminuem a probabilidade de existirem vulnerabilidades no sistema. Sendo assim, o objetivo deste projeto é descrever um estudo de caso que consiste em identificar, descrever e comparar o estado da prática na utilização de técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas em três organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis.

Financiamento: -.

Palavras-chave: Desenvolvimento de software, Engenharia de Software, Métodos Ágeis, Segurança de Software.

Resumo:

Métodos ágeis foram criados para sanar fraquezas reais e perceptíveis dos

métodos de desenvolvimento de software tradicionais. Entretanto, pela pressão na entrega de um produto de software dentro do prazo estimado, requisitos de segurança são pouco mensurados ou até deixado de lado. Durante o desenvolvimento ágil de software é importante detectar possíveis vulnerabilidades durante seu ciclo de vida, utilizando processos, atividades ou ferramentas.

Esse projeto descreve um instrumento a ser utilizado em um estudo exploratório e qualitativo em três organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis. A partir dos resultados dessa pesquisa, espera-se identificar as dificuldades e oportunidades para conciliar métodos ágeis e segurança de software, o nível de aplicação das técnicas e ferramentas e as habilidades e necessidades de formação da equipe em segurança de software nas três organizações estudadas.

Espera-se que o instrumento a ser aplicado no estudo de caso seja útil para pesquisadores e empresas interessadas em avaliar a utilização de técnicas e ferramentas de segurança no contexto de métodos ágeis.

Introdução:

Agilidade tornou-se a palavra principal quando é descrito um moderno processo de desenvolvimento de software (SOMMERVILLE, 2011). Constantes mudanças podem ocorrer durante o desenvolvimento de software como: mudanças no software que está sendo criado, mudanças nos membros da equipe, mudanças devido a novas tecnologias ou mudanças no projeto que cria o produto.

No desenvolvimento ágil, um produto de software é desenvolvido por indivíduos trabalhando em equipes e que cada membro tem uma habilidade específica, as quais colaboram com sucesso do projeto (HOWARD;LIPNER, 2009).

Esse modelo de desenvolvimento prioriza a satisfação dos clientes e a entrega incremental de produto de software por meio de equipes de projetos pequenas e altamente motivadas, artefatos de engenharia de software mínimos e simplicidade no desenvolvimento (BECK, et.al., 2001).

Para Pressman (2009) o desenvolvimento ágil oferece benefícios importantes, no entanto, não é indicado para todos os tipos de projetos, softwares e situações. Para utilizá-lo, é essencial que a equipe adapte os processos e atividades existentes e mantenha apenas artefatos realmente essenciais, utilizando processos enxutos, enfatizando a estratégia de entrega incremental.

Um outro aspecto importante é o conceito de vulnerabilidade. Uma vulnerabilidade em software é um conjunto de condições que podem levar à violação de uma política de segurança (SEACORD;HOUSEHOLDER, 2005). Tais condições podem ser oriundas da má especificação de requisitos de segurança, problemas com o *design*, práticas de codificação insegura, falhas nas atividades de garantia de segurança ou problemas de manutenção do sistema (HOWARD;LIPNER, 2009).

A vulnerabilidade é uma fraqueza na aplicação, o que permite que um atacante cause danos aos *stakeholders* de um sistema (OWASP, 2010b). Em um sistema, procedimentos de segurança, controles internos ou de implementação que podem ser explorados por ameaças são considerados vulnerabilidades (NIST, 2009).

Assim, é de extrema importância detectar vulnerabilidades durante o desenvolvimento de software, pois informações perdidas, utilizadas incorretamente e acessadas por pessoas não autorizadas podem prejudicar uma organização.

Alguns termos correlatos vêm juntamente a este, como é o caso de ameaças (GRIESKAMP et. al., 2004). Uma ameaça é uma ocorrência que pode acarretar algum dano para o sistema ou organização como fraqueza em um ativo ou grupos de ativos (PINHEIRO, 2011).

Um ataque ocorre quando uma ameaça intencional é realizada por motivos variados, como: ganhos financeiros, venda de informações, espionagem ou sabotagem (PINHEIRO, 2011). Outro termo associado a vulnerabilidades é o risco, o qual pode ser definido como a probabilidade da ocorrência de uma ameaça (VERDON;MCGRAW, 2004).

Uma vulnerabilidade também está ligada ao termo falha que pode ser definido como a incapacidade de executar uma função requerida dentro dos limites especificados (GROUP, 2010). É possível associar uma vulnerabilidade ao termo malicioso, o qual pode ser qualquer código adicionado, alterado ou removido de um sistema de software, a fim de causar intencionalmente danos ou subverter a função pretendida do sistema (MCGRAW;MORRISETT, 2000).

Um defeito pode ser introduzido em qualquer uma das fases do ciclo de desenvolvimento de software, como na especificação de requisitos, *design*, implementação, verificação ou manutenção (JONES, 2013).

Dessa forma, para prevenir a ocorrência de vulnerabilidades, é recomendável utilizar técnicas e ferramentas de segurança desde as primeiras fases do ciclo de vida (HOWARD;LIPNER, 2009).

Hipótese: Técnicas e ferramentas para detecção de vulnerabilidades são utilizadas no desenvolvimento ágil de software.

Objetivo Primário: O objetivo geral deste estudo consiste em identificar, descrever e comparar o estado da prática na utilização de técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas em três organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis.

Técnicas e Ferramentas avaliadas: Os processos de desenvolvimento de software seguro estudados são:

Table 1: Processos de segurança obtidos pela revisão bibliográfica

OWASP CLASP	Patrocinado pela Open Web Application Security Project (OWASP), CLASP é projetado para ajudar equipes de desenvolvimento de software a construírem a segurança desde as fases iniciais do ciclo de vida de desenvolvimento de software de forma estruturada, mensurável e repetível.
MC Graw	É um processo de segurança que abrange segurança em todo ciclo de vida de desenvolvimento de software. Centra-se na incorporação de requisitos de segurança, modelagem de *abuse cases*, especificação de requisitos de segurança, análise de riscos, *misuse cases* e uso de boas práticas.
MS SDL	O Ciclo de Vida de Desenvolvimento de Segurança da Microsoft tem a participação do cliente na fase de análise de requisitos a fim de identificar os objetivos de segurança. O especialista em segurança atribuído ao projeto especifica critérios de saída para cada fase de desenvolvimento.
SSDM	Modelo de Segurança no Desenvolvimento de Software é um processo que incorpora atividades de segurança no ciclo de vida de desenvolvimento de software cascata. Realiza modelagem de riscos e modelagem de ameaças, lista as possíveis vulnerabilidades e define políticas de segurança.
TSP Secure	Processo para desenvolvimento de software seguro em equipe é um processo que tem por objetivo orientar as equipes no desenvolvimento de produtos de software seguro e confiável.
Howard e Lipner	Propõem a incorporação de atividades de segurança aos Métodos Ágeis. É necessário ter atenção, pois algumas atividades podem levar a um processo não ágil.

Porém, destes, apenas as técnicas e ferramentas adotadas nos processos OWASP CLASP, de Mc Graw e de Howard e Lipner serão identificadas, descritas e comparadas.

Metodologia proposta:

Será realizada uma pesquisa exploratória e qualitativa, composta por entrevistas relacionadas às técnicas e ferramentas para detecção de vulnerabilidades adotadas identificadas por meio de uma revisão bibliográfica. A pesquisa utilizará um instrumento para coleta de informações e que orientará a aplicação de questionários e a utilização da técnica de análise de conteúdo.

Para o desenvolvimento do instrumento deste estudo de caso a seguinte estratégia foi adotada:

1- Seleção das técnicas e ferramentas para detecção de vulnerabilidades utilizadas em comum pelos processos OWASP CLASP, Howard e Lipner e de McGraw, ou seja, as técnicas e ferramentas dos processos mais utilizados e pesquisados pelos autores.

2- A partir da seleção dos três processos de desenvolvimento seguro de software, foram identificadas e selecionadas as técnicas e ferramentas de segurança comumente aplicadas pelos seguintes processos: OWASP CLASP, Howard e Lipner e de McGraw, a fim de comparar e listar os benefícios que foram relatados na literatura para essas técnicas e ferramentas.

O resultado das etapas 1 e 2 está na Tabela 2 na qual as técnicas e ferramentas foram classificadas em: concepção, modelagem de ameaças e gerenciamento de riscos, *design* e codificação segura, ferramentas de segurança, testes de segurança e distribuição.

Table 2: Técnicas e ferramentas para detecção de vulnerabilidades utilizadas pelos processos de segurança

Concepção	**Design e codificação segura**
Existe um especialista em segurança	Design de segurança
Estabelece requisitos de segurança	Codificação segura
Escreve abuse/misuse cases/stories	
Modelagem de ameaças e gestão de riscos	**Ferramentas de segurança**
Modelagem de ameaças	Ferramenta de modelagem de ameaças/riscos
Técnicas de contramedidas de segurança	Ferramenta de análise dinâmica de código
Modelagem de riscos	Ferramenta de análise estática de código
	Ferramenta de revisão de código
Distribuição	**Testes de segurança**
Gestão de respostas a incidentes	Avaliação de vulnerabilidades
	Teste de penetração
	Fuzz testing
	Teste baseado em risco
	Revisão de segurança de código

A partir da classificação das técnicas e ferramentas listadas na Tabela 2, foram identificadas questões associadas à implantação dessas técnicas por empresas desenvolvedoras de software e os seu benefícios, propostos pela literatura para cada uma delas. A Tabela 3, descrita no Apêndice A.3, apresenta as questões e os benefícios identificados para todas técnicas e ferramentas selecionadas na Tabela 2.

Serão convidadas três organizações desenvolvedoras de software que utilizam métodos ágeis para participar do experimento. Tais organizações estão sendo definidas, bem como segmento e número de participantes. Será apresentada as

técnicas e ferramentas de segurança da Tabela 2. Os participantes do experimento serão convidados pessoalmente. Aos que aceitarem participar, retornaremos o contato com todas as informações necessárias para realizá-lo.

O instrumento será implementado por meio de um questionário disponibilizado no Google *Forms* e as entrevistas serão realizadas por Skype, caso não haja possibilidade de serem presenciais. O instrumento é composto por questões que avaliam uma técnica ou ferramenta de detecção de vulnerabilidade. O instrumento completo poderá ser encontrado no Apêndice A.

A Tabela 3, do Apêndice A.3, apresenta as questões desenvolvidas no instrumento para avaliar o grau de implantação das técnicas e ferramentas, bem como os benefícios percebidos pelos desenvolvedores. A forma de avaliação das técnicas e ferramentas é descrita a seguir.

Cada pergunta da Tabela 3 possui um peso de 0,25 na porcentagem total de implantação de uma técnica ou ferramenta. Portanto, a porcentagem de implantação é calculada da seguinte forma:

*%IMP Processo = ((P1 *0,25) + (P2 *0,25) + (P2 *0,25) + (P4 *0,25))*

onde *%IMP* é a porcentagem de implantação da técnica ou ferramenta e *P1* – Pergunta 1, *P2* – Pergunta 2, *P3* – Pergunta 3 e *P4* – Pergunta 4 são os valores das respostas dos participantes. As repostas variam de 0% (não implantado) até 100% (completamente implantado).

Na Tabela 3, observa-se na linha dois a técnica *requisitos de segurança*. A essa técnica estão relacionadas quatro perguntas (*P1* a *P4*) que descrevem o grau de implantação da técnica. Considera-se que os sujeitos tenham respondido "Sim" para todas as quatro questões relacionadas à implantação da técnica de requisitos de segurança. Logo, o resultado do cálculo da implantação relacionada a pergunta será:

%IMP Processo = Soma ((1 *0,25) + (1 *0,25) + (1 *0,25) + (1 *0,25))

Portanto, a porcentagem de implantação da técnica de *requisitos de segurança* é de 100%.

Em relação aos benefícios previstos pela literatura para a técnica, supôs-se que três dos sujeitos da pesquisa tenham respondido "Sim" para as quatro questões propostas, sendo apenas uma resposta "Não" para as quatro questões. Logo, a porcentagem de benefícios obtidos na utilização da técnica de *requisitos de segurança* pela organização é de 75%. Esse cálculo foi realizado pela média da porcentagem obtida pela resposta dos sujeitos. O sujeito também irá apontar quais benefícios foram obtidos, bem como outros benefícios que podem ter sidos obtidos e não estão relatados na literatura.

O exemplo acima analisou apenas a porcentagem do grau de benefício e de implantação da técnica em apenas uma organização. Foram confrontados os benefícios previstos na literatura com os benefícios obtidos com a implantação pela organização. Nesta, o grau de implantação foi alto, 100% assim como seu grau de benefícios 75%. Sendo assim, a técnica de requisitos de segurança teve alto grau de benefícios em correspondência a um alto grau de implantação.

Dessa maneira, o instrumento do experimento permitirá relacionar a percepção dos benefícios obtidos com as técnicas e ferramentas e sua implantação para cada organização.

O instrumento também prevê avaliar as habilidades e necessidade de formação da equipe como na Seção A.3.3 do Apêndice A. Para essas questões será utilizada a escala de Likert de 5 pontos, onde: 1 =Baixa, 2 =Básica, 3= Média, 4 = Alta e 5 = Especialista. Além disso, será fornecida uma breve explicação de cada termo utilizado na pesquisa para os respondentes, conforme mostra a Tabela 3.

Table 3: Escala de níveis de habilidades nas técnicas e ferramentas de segurança de software

Escala	Descrição dos níveis de habilidades
Baixo	Não possui experiência de trabalho nesta área.
Básico	Possui nível de experiência adquirida em cursos. Necessita de ajuda ao executar as tarefas.
Moderado	É capaz de completar com sucesso as tarefas nesta área. Pode solicitar o auxílio de um especialista, mas consegue executar as habilidades de forma independente.
Alto	É capaz de executar as tarefas associadas à área sem auxílio de terceiros. Pode ser reconhecido dentro da organização como um consultor imediato.
Especialista	É conhecido como um especialista. Possui capacidade de fornecer orientação, solucionar problemas e responder a perguntas relacionadas com esta área.

A análise será baseada nos papéis dos membros dentro da equipe ágil de cada organização. Assim, os sujeitos do estudo de caso serão: analistas de requisitos, arquitetos de software, desenvolvedores e testadores. A quantidade de sujeitos ainda será definida.

Dessa maneira, a metodologia utilizada nesta pesquisa é o estudo de caso múltiplo. Baseado nesta estratégia de múltiplos casos, os estudos desta pesquisa serão realizados em três organizações que utilizam métodos ágeis como processo de desenvolvimento de software.

De acordo com Yin (2013) cinco componentes são importantes em um projeto de pesquisa que utiliza estudo de caso, são eles:

- Questões de estudo.
- Proposições.
- Unidade de análise.
- Lógica que une os dados às proposições.
- Critérios de interpretação dos resultados.

1. Questões de estudo: as questões a serem respondidas devem ser do tipo "como" e "por quê", as quais têm a função de esclarecer a natureza do estudo. Estas devem estar em conformidade com os objetivos e questões da pesquisa As questões encontram-se no Apêndice A.

2. Proposições de estudo: cada proposição dirige a atenção para algo que deve ser examinado no âmbito do estudo. Ou seja, as proposições devem ser analisadas a partir das questões de pesquisa e objetivos, como por exemplo, identificar as técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas pelas organizações brasileiras desenvolvedoras de software.

3. Unidade de análise: define o "caso" a ser estudado. Nesta pesquisa, o "caso" são as organizações brasileiras desenvolvedoras de software que aplicam métodos ágeis e participaram do estudo de caso.

4. Lógica que une os dados às proposições: consiste na análise de dados obtidas no estudo de caso como combinações de padrões, construção de explicação, análise de séries temporais, modelos lógicos, e síntese de casos cruzados. Neste estudo de caso, a lógica é dada pela metodologia de análise do grau de implantação e dos benefícios percebidos e também pela análise do conteúdo das respostas dos sujeitos.

5. Critérios de interpretação dos resultados: consiste na adoção de técnicas estatísticas para interpretações e organizações das análises de dados, bem como adoção de gráficos, tabelas, quadros, entre outros. Será utilizada uma estrutura de quadro para apresentação dos resultados e sua interpretação. A Figura 2 apresenta um exemplo de um dos quadros de resultados.

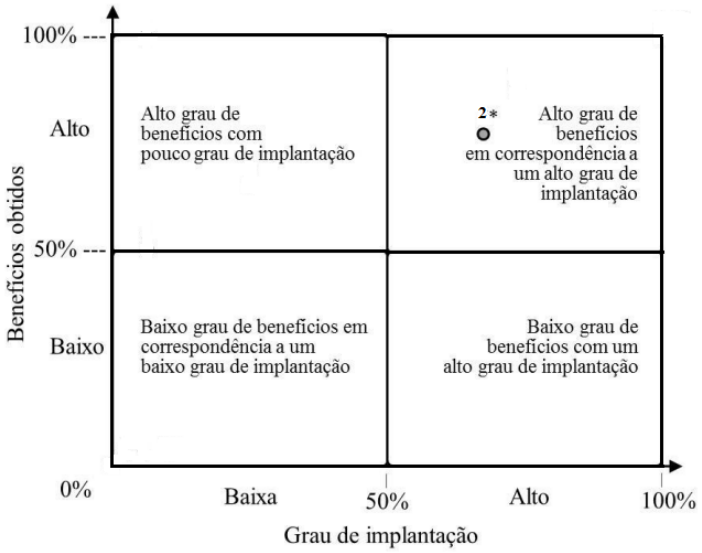

Figure 1: Benefícios obtidos e grau de implantação

Recomenda-se que um projeto de pesquisa seja estruturado por meio desses cinco componentes, a fim de auxiliar na coleta, análise e conclusão dos resultados obtidos.

Riscos: Não há riscos físicos a saúde dos participantes. Sendo assim, um dos principais cuidados é garantir o sigilo dos participantes, confiabilidade do conteúdo, validade de conteúdo, validade de construção, validade externa.

Benefícios: Espera-se obter a relação das técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas nas organizações brasileiras analisadas, bem como o motivo da escolha dessas técnicas e ferramentas.

Além disso, espera-se obter as dificuldades e limitações da utilização das técnicas e ferramentas para detecção de vulnerabilidades em desenvolvimento ágil de software, lições aprendidas e os benefícios esperados e obtidos na sua utilização.

Desfecho Primário: Este projeto tem o objetivo de descrever o estudo de caso múltiplo que será realizado em três organizações desenvolvedoras de software que utilizam métodos ágeis. O estudo visa caracterizar a utilização de técnicas e ferramentas para detecção de vulnerabilidade em organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis. Assim, foram descritos brevemente os principais conceitos associados a métodos ágeis e vulnerabilidades, bem como o instrumento a ser aplicado.

Detectar vulnerabilidades em desenvolvimento ágil é uma atividade que requer não apenas a utilização de ferramentas e técnicas no final do desenvolvimento, mas principalmente a adoção de práticas de segurança ao longo do ciclo de vida de desenvolvimento de software. O estudo de caso descrito neste projeto visa indicar o estado atual do uso dessas técnicas e ferramentas em organizações brasileiras e o benefícios percebidos pela sua adoção.

A partir desse diagnóstico, será possível identificar as dificuldades e oportunidades para conciliar métodos ágeis e segurança de software, o nível de aplicação e os benefícios dessas técnicas e ferramentas, bem as habilidades e necessidades de formação da equipe em segurança de software nas três organizações estudadas.

Espera-se que os resultados do estudo de caso sejam úteis ao planejamento de políticas para a implementação de abordagens de segurança em organizações que utilizam métodos ágeis no Brasil.

Tamanho da Amostra no Brasil: Três organizações desenvolvedoras de software que utilizam métodos ágeis. O número de sujeitos participantes do experimento do estudo de caso ainda será definido. Os sujeitos serão classificados em: analistas de requisitos, arquitetos de software, desenvolvedores e testadores.

Referências

BECK, K. Extreme programming explained: Embrace change. 2a. ed. [S.l.]: Addison-Wesley, Upper Saddle River, NJ, 2000. 37–51 p.

HOWARD, M.; LIPNER, S. The security development lifecycle: SDL, a process for developing demonstrably more secure software. 1a. ed. [S.l.]: Microsoft Press Redmond, WA, USA, 2006. 225—239 p.

MCGRAW, G. Software Security. IEEE Security & Privacy, IEEE, v. 2, n. 2, p. 80–83, 2004. Citado na página 49. MCGRAW, G.; MORRISETT, G. Attacking malicious code: A report to the infosec research council. IEEE software, IEEE Computer Society, v. 17, n. 5, p. 1–11, 2000.

MICHAELIS. Dicionário brasileiro da língua portuguesa: Segurança. 2016. Disponível em: http://michaelis.uol.com.br/palavra=segurança.

NIST. National vulnerability database. 2009. Disponível em: hhttps://www.-nist.gov/programs-projects/national-vulnerability-database-nvdi.

PINHEIRO, J. M. dos S. Amea¸cas e ataques aos sistemas de informação: Prevenir e antecipar threats and attacks to information systems: Prevent and antecipate. Cadernos UniFOA Centro Universitário de volta redonda, Fundação Oswaldo Aranha, UniFOA, v. 5, p. 1–11, 2011.

PRESSMAN, R. Engenharia de Software. 7a. ed. [S.l.]: McGraw Hill, Brasil, 2009. 81–106p.

OWASP. CLASP Concepts. 2010. Disponível em: hhttps://www.owasp.org/ index.php/ CLASP Concept.

OWASP. OWASP Top Ten. 2010. Disponível em: https://www.owasp.org/index.php/Category: OWASP Top Ten Project.

OYETOYAN, T. D.; CRUZES, D. S.; GILJE, M. J. An empirical study on the relationship between software security skills, usage and training needs in agile settings. Conference Publishing Services (CPS), p. 548–555, 2016.

SEACORD, R. C.; HOUSEHOLDER, A. D. A structured approach to classifying security vulnerabilities. Carnegie Mellon University, Software Engineering Institute, Carnegie Mellon University, p. 12–16, 2005.

SOMMERVILLE, I. Software Egineering. 9a. ed. [S.l.]: Pearson Education, Boston, Massachusetts, 2011. 56–81p.

YIN, R. K. Case study research: Design and methods. 5a. ed. [S.l.]: Sage publications, 2013. 25–95 p.

A Apêndice

A.1 Um estudo de caso sobre o uso de técnicas e ferramentas para detecção de vulnerabilidades em organizações que utilizam métodos ágeis

Esse *survey* tem por objetivo entender e perceber as necessidades de treinamento dos membros da equipe em adquirir conhecimento, competência em segurança, identificar as áreas onde as equipes estão (são) propensas a adotar certamente as técnicas e ferramentas para detecção de vulnerabilidades, estimando os interesses e aptidões dos indivíduos, estabelecer o nível de segurança de cada recurso disponível dentro das equipes e a documentação em segurança softwares existentes.

Finalmente, o *survey* irá contribuir com a cultura de segurança de software dentro da organização por meia da consciência do aprendizado contínuo. Ele está dividido em cinco seções. A Seção A aborda informações gerais sobre a organização. A Seção B foca em informações gerais sobre os projetos. Já a Seção C relaciona a aptidões, interesses e uso atual de técnicas e ferramentas de segurança de software; a Seção D está focada nas necessidades de treinamento e, finalmente, a Seção E são perguntas gerais sobre a implantação.

A.2 Instrumento do estudo de caso realizado na pesquisa

A.2.1 Seção A Informações gerais sobre a organização

As perguntas a seguir foram aplicadas nas entrevistas com os envolvidos por parte da equipe. O objetivo dessas perguntas é levantar informações sobre as organizações estudadas. As perguntas aplicadas foram:

1) Qual o número de funcionários na organização?
2) Qual o faturamento médio da organização?

- Até 500 mil
- De 500 mil a 1 milhão
- De 1 milhão a 3 milhões
- Acima de 3 milhões
- Não divulgado.

3) Qual a área de atuação da organização?

- Agricultura
- Industrial
- Eletricidade e gás
- Água, esgoto, atividades de gestão de resíduos

- Construção
- Comércio
- Transportes
- Alimentos
- Informação e comunicação
- Financeiro
- Imobiliário
- Administrativa
- Pública
- Educação
- Saúde
- Artes, cultura, esporte e recreação
- Outras atividades de serviços
- Serviços domésticos
- Órgãos internacionais

4) Quais tipos de produtos são comercializados pela organização?
5) Qual o número de colaboradores da área de TI da organização?

A.2.2 Seção B Informações gerais sobre os projetos

As perguntas a seguir foram aplicadas nas entrevistas com os envolvidos por parte da equipe. O objetivo dessas perguntas é levantar informações sobre o projeto de implantação de técnicas e ferramentas para detecção de vulnerabilidades nas organizações estudadas.

6) Qual sua função na equipe de desenvolvimento ágil?

- Analista
- Arquiteto de software
- Desenvolvedor
- Testador
- Product Owner
- Scrum Master
- Outro (especificar)

7) Qual (is) método (s) ágil (is) você utiliza?

- Scrum
- Extreme Programming (XP)
- Feature Driven Developlment (FDD)
- Lean Software Development
- Crystal Methods
- Kanban
- Agile Unifield Process (AUP)
- Dynamic Systems Development Method (DSDM)
- Outro (especificar)

8) Você possui experiência em segurança de software?

- Sim
- Não

9) Nº de anos com desenvolvimento de software?
10) Qual o nome do produto?
11) Tipo de produto (ex: *web*, *mobile*, redes, *e-commerce*, etc.)

- Web
- Mobile
- Rede
- E-commerce
- Outro (especificar)

12) Quem são os principais *stakeholders* do projeto?
13) Qual o tamanho da equipe de consultoria que implantou as técnicas e ferramentas de segurança no projeto?
14) Adotou-se alguma metodologia de implantação?

- PMI
- Scrum
- Metodologia própria
- Outra (especificar)

15) Responsável pelo desenvolvimento de atividades de segurança:

- Especialista em segurança
- Analista
- Arquiteto
- Desenvolvedor
- Testador
- Outros (especificar)

A.2.3 Seção C: Aptidão, interesses e uso atual

As perguntas a seguir foram aplicadas nas entrevistas com os envolvidos por parte do projeto. O objetivo dessas perguntas é levantar informações sobre as aptidões, interesses e uso atual de técnicas e ferramentas para detecção de vulnerabilidades. Essa Seção utilizará a escala abaixo:

Escala	Descrição dos níveis de habilidades
Baixo	Não possui experiência de trabalho nesta área.
Básico	Possui nível de experiência adquirida em cursos. Necessita de ajuda ao executar as tarefas.
Moderado	É capaz de completar com sucesso as tarefas nesta área. Pode solicitar o auxílio de um especialista, mas consegue executar as habilidades de forma independente.
Alto	É capaz de executar as tarefas associadas à área sem auxílio de terceiros. Pode ser reconhecido dentro da organização como um consultor imediato.
Especialista	É conhecido como um especialista. Possui capacidade de fornecer orientação, solucionar problemas e responder a perguntas relacionadas com esta área.

Figure 2: Escala de habilidades

16) Qual seu nível de interesse nessa técnica ou ferramenta?

Especialista em segurança
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Requisitos de segurança
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Abuse/Misuse Case/Stories
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Modelagem de ameaças
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Modelagem de riscos
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Gestão de respostas a incidentes
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Design de segurança
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Contramedidas de segurança
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Codificação segura
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Ferramenta de modelagem de ameaças/riscos
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Ferramenta de análise estática de código
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Ferramenta de análise dinâmica de código
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Ferramenta de revisão de código
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Avaliação de vulnerabilidades
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Teste de penetração
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Fuzz Testing
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Teste baseado em risco
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

Revisão de segurança de código
Não interessado/Levemente interessado/Moderadamente interessado/Muito interessado/Não sei.

17) Se você atualmente faz alguma atividade ou usa alguma ferramenta na sua função, por favor assinale sim, caso contrário não:

Requisitos de segurança
Sim/Não.

Abuse/Misuse Case/Stories
Sim/Não.

Modelagem de ameaças
Sim/Não.

Modelagem de riscos
Sim/Não.

Gestão de respostas a incidentes
Sim/Não.

Design de segurança
Sim/Não.

Contramedidas de segurança
Sim/Não.

Codificação segura
Sim/Não.

Ferramenta de modelagem de ameaças/riscos
Sim/Não.

Ferramenta de análise estática de código
Sim/Não.

Ferramenta de análise dinâmica de código
Sim/Não.

Ferramenta de revisão de código
Sim/Não.

Avaliação de vulnerabilidades
Sim/Não.

Teste de penetração
Sim/Não.

Fuzz Testing
Sim/Não.

Teste baseado em risco
Sim/Não.

Revisão de segurança de código
Sim/Não.

A.2.4 Seção D: Treinamento

As perguntas a seguir foram aplicadas nas entrevistas com os envolvidos por parte da equipe. O objetivo dessas perguntas é levantar informações em relação as necessidades de treinamento da equipe em técnicas e ferramentas para detecção de vulnerabilidades nas organizações estudadas.

18) Por favor responda sim para a técnica ou ferramenta que gostaria de receber treinamento.

Especialista em segurança
Sim/Não.

Requisitos de segurança
Sim/Não.

Abuse/Misuse Case/Stories
Sim/Não.

Modelagem de ameaças
Sim/Não.

Modelagem de riscos
Sim/Não.

Gestão de respostas a incidentes
Sim/Não.

Design de segurança
Sim/Não.

Contramedidas de segurança
Sim/Não.

Codificação segura
Sim/Não.

Ferramenta de modelagem de ameaças/riscos
Sim/Não.

Ferramenta de análise estática de código
Sim/Não.

Ferramenta de análise dinâmica de código
Sim/Não.

Ferramenta de revisão de código
Sim/Não.

Avaliação de vulnerabilidades
Sim/Não.

Teste de penetração
Sim/Não.

Fuzz Testing
Sim/Não.

Teste baseado em risco
Sim/Não.

Revisão de segurança de código
Sim/Não.

19) Qual seu nível de conhecimento em:

Requisitos de segurança
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Abuse/Misuse Case/Stories
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Modelagem de ameaças
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Modelagem de riscos
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Gestão de respostas a incidentes
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Design de segurança
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Contramedidas de segurança
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Codificação segura
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Ferramenta de modelagem de ameaças/riscos
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Ferramenta de análise estática de código
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Ferramenta de análise dinâmica de código
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Ferramenta de revisão de código
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Avaliação de vulnerabilidades
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Teste de penetração
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Fuzz Testing
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Teste baseado em risco
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Revisão de segurança de código
0- Não sei/1-Baixo/2-Básico/3-Médio/4-Alto/5-Especialista.

Benefícios da implantação de técnicas e ferramentas para detecção de vulnerabilidades

Para cada técnica e ferramenta foram elaboradas quatro perguntas, as quais o entrevistado pode atribuir o valor de implantação entre 0 e 1, ou seja, de 0% até 100%. Cada pergunta tem peso de 0,25 na porcentagem total de implantação de uma técnica ou ferramenta. Portanto, a porcentagem de implantação de uma

técnica ou ferramenta é calculada da seguinte forma:

> %IMP Processo = Soma ((P1 *0,25) + (P2 *0,25) + (P2 *0,25) + (P4 *0,25))

onde *%IMP* é a porcentagem de implantação da técnica ou ferramenta e *P1* – Pergunta 1, *P2* – Pergunta 2, *P3* – Pergunta 3 e *P4* – Pergunta 4 são os valores das respostas dos participantes. As respostas variam de 0% (não implantado) até 100% (completamente implantado).

Na Tabela 3, observa-se na linha dois a técnica requisitos de segurança. A essa técnica estão relacionadas quatro perguntas (*P1* a *P4*) que descrevem o grau de implantação da técnica. Considera-se que os sujeitos tenham respondido "Sim" para todas as quatro questões relacionadas à implantação da técnica de requisitos de segurança. Logo, o resultado do cálculo da implantação relacionada a pergunta será:

> %IMP Processo = Soma ((1 *0,25) + (1 *0,25) + (1 *0,25) + (1 *0,25))

Portanto, é possível considerar que a porcentagem de implantação da técnica de requisitos de segurança é de 100%.

Em relação aos benefícios previstos pela literatura para a técnica de requisitos de segurança, foi suposto que três dos sujeitos da pesquisa tenham respondido "Sim" para as quatro questões propostas, sendo apenas uma resposta "Não" para as quatro questões.

Assim é possível considerar que a porcentagem de benefícios obtidos na utilização da técnica de requisitos de segurança pela organização é de 75%. Esse cálculo foi realizado pela média da porcentagem obtida pela resposta dos sujeitos.

O sujeito também irá apontar quais benefícios foram obtidos, bem como, outros benefícios que podem ter sidos obtidos e não estão relatados na literatura.

O exemplo acima analisou apenas a porcentagem do grau de benefício e de implantação da técnica de requisitos de segurança em apenas uma organização. Foram confrontados os benefícios previstos na literatura com os benefícios obtidos com a implantação pela organização. Nesta, o grau de implantação foi alto, 100% assim como seu grau de benefícios 75%. Sendo assim, a técnica de requisitos de segurança teve alto grau de benefícios em correspondência a um alto grau de implantação.

Sendo assim, o instrumento do experimento permitirá relacionar a percepção dos benefícios obtidos com as técnicas e ferramentas e sua implantação para cada organização. Os resultados obtidos serão relacionados à ocorrência de benefícios ou não da adoção de determinada técnica e ferramenta de segurança. O entrevistado também irá apontar quais benefícios foram obtidos, bem como outros benefícios que podem ter sidos obtidos e não estão relatados na literatura.

A.3 Instrumento do estudo de caso

Table 4: Instrumento do estudo de caso.

Nº	Técnicas e Ferramentas	Perguntas	% IMP Pergunta	%IMP Tec. e Ferramentas.	Benefícios Previstos	Benefícios Obtidos
1	Especialista em Segurança	1) Existe algum especialista em segurança de software? 2) Foram definidas políticas de segurança para o desenvolvimento ágil de software? 3) Foram definidas responsabilidades de aplicação de técnicas e ferramentas de segurança? 4) As politicas de segurança são aplicadas no projeto?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Especificar políticas de segurança de software para o projeto. 2) Aplicar e acompanhar critérios de políticas de segurança durante o projeto. 3) Melhorar a aplicação e controle de segurança no desenvolvimento de ágil de software. 4) Controlar a aplicação das políticas de segurança e mantê-la nos níveis adequados de segurança.	S/N S/N S/N S/N
2	Requisitos de segurança	1) São especificados requisitos de segurança para a aplicação? 2) Existe algum procedimento para gerenciar esses requisitos? 3) Foi realizada alguma priorização de requisitos de segurança? 4) É utilizada alguma técnica para a coleta de requisitos de segurança?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Identificar os objetivos e restrições de segurança na aplicação. 2) Facilitar a organização e classificação dos requisitos de segurança a fim de manipulá-los quando necessário. 3) Identificar os requisitos mais críticos de segurança. 4) Facilitar a coleta e entendimento dos requisitos de segurança.	S/N S/N S/N S/N
3	*Abuse/misuse cases/stories*	1) São especificados requisitos de segurança utilizando *abuse/misuse cases?* 2) Utiliza algum procedimento para a especificação de requisitos com *abuse/misuse cases*? 3) Foi realizada alguma priorização de *abuse/misuse cases*? 4) São utilizadas ferramentas para a criação dos *misuse/abuse cases*?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Descrever os comportamentos prejudiciais e indesejáveis do sistema. 2) Facilitar a identificação dos possíveis comportamentos prejudiciais e indesejáveis do sistema. 3) Identificar os requisitos mais críticos de segurança. 4) Facilitar o entendimento e difundir as informações obtidas através dos *abuse/misuse cases/stories* à equipe.	S/N S/N S/N S/N
4	Modelagem de ameaças	1) Os requisitos mais críticos são modelados, em relação às possíveis ameaças? 2) São identificadas a probabilidade e impacto das ameaças à segurança? 3) Utiliza alguma estrutura para a modelagem de ameaças? 4) Alguma documentação para modelagem de ameaças é utilizada?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Identificar as ameaças mais críticas e suas contramedidas. 2) Identificar as ameaças de segurança mais críticas, probabilidades e seus efeitos. 3) Identificar e facilitar o entendimento em relação as possíveis ameaças na aplicação. 4) Reportar à equipe a cerca das possíveis ameaças na aplicação e suas contramedidas.	S/N S/N S/N S/N

5	Modelagem de riscos	1) Existe alguma priorização de requisitos na análise de riscos? 2) São identificadas a probabilidade e impacto dos ataques à segurança? 3) Utiliza alguma estrutura para a análise de riscos? 4) Alguma documentação para a análise de riscos é utilizada?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Facilitar a identificação dos riscos nas funcionalidades mais críticas e suas mitigações. 2) Facilitar a identificação dos riscos de segurança mais críticos, probabilidades e seus efeitos. 3) Identificar e facilitar o entendimento em relação aos possíveis riscos de falta de segurança na aplicação. 4) Reportar à equipe a cerca dos possíveis riscos na aplicação e suas mitigações.	S/N S/N S/N S/N
6	Gestão de respostas a incidentes	1) Existe o gerenciamento de incidentes de segurança? 2) São identificados e listados os possíveis incidentes? 3) Para cada incidente identificado, existem respostas? 4) É utilizada alguma documentação específica para essa gestão?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Facilitar a identificação das ameaças mais críticas e suas contramedidas. 2) Facilitar a identificação das ameaças de segurança mais críticas, probabilidades e seus efeitos. 3) Identificar e facilitar o entendimento em relação as possíveis ameaças na aplicação. 4) Reportar à equipe a cerca das possíveis ameaças na aplicação e suas contramedidas.	S/N S/N S/N S/N
7	*Design* de segurança	1) Realiza a identificação de componentes críticos da aplicação? 2) São aplicadas políticas de segurança no *Design* da aplicação? 3) Utiliza a modelagem de ameaças e riscos para se basear no *Design*? 4) É utilizada alguma notação específica para *Design* seguro?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Facilitar a identificação dos componentes críticos da aplicação. 2) Aplicar critérios de segurança nos componentes da aplicação. 3) Facilitar a aplicação de critérios de segurança através da utilização da modelagem de ameaças e riscos. 4) Facilitar o entendimento do *Design* em relação a segurança.	S/N S/N S/N S/N
8	Contramedidas de segurança	1) Existe o gerenciamento dos incidentes de segurança? 2) São identificados e listados os possíveis incidentes de segurança? 3) Para cada incidente identificado, são identificadas as contramedidas? 4) É utilizada alguma documentação específica para o gerenciamento de contramedidas?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Organizar as contramedidas de segurança para cada ameaça identificada. 2) Introduzir proteção contra possíveis ameaças. 3) Facilitar a identificação de contramedidas de segurança para cada ameaça identificada. 4) Difundir a informação relacionada às ameaças e suas contramedidas para a equipe.	S/N S/N S/N S/N
9	Codificação segura	1) Utiliza práticas de codificação segura? 2) Utiliza técnicas de revisão de código? 3) Aplica políticas de segurança durante a codificação? 4) Utiliza ferramentas de segurança durante a codificação?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Difundir a segurança na codificação, através da utilização de compiladores recentes e organizando listas com funções inseguras, por exemplo. 2) Analisar se as práticas de codificação segura vêm sendo aplicadas. 3) Assegurar se as práticas de segurança que foram definidas inicialmente vêm sendo aplicadas. 4) Facilitar a aplicação de práticas de codificação segura.	S/N S/N S/N S/N

10	Ferramenta de modelagem de ameaça/risco	1) São utilizadas ferramentas para modelar as possíveis ameaças? 2) São utilizadas ferramentas para calcular a probabilidade e impacto das ameaças? 3) Utiliza alguma ferramenta para gerenciar as ameaças? 4) As ferramentas utilizadas identificam as contramedidas e mitigações?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Facilitar a identificação das ameaças mais críticas e suas contramedidas. 2) Facilitar a identificação das ameaças de segurança mais críticas, probabilidades e seus efeitos. 3) Identificar e facilitar o entendimento em relação as possíveis ameaças na aplicação. 4) Reportar à equipe a cerca das possíveis ameaças na aplicação e suas contramedidas.	S/N S/N S/N S/N
11	Ferramenta de análise estática de código	1) São utilizadas ferramentas para analisar defeitos estaticamente no código-fonte? 2) Tais ferramentas reportam os possíveis defeitos? 3) Identificam quais trechos de código são mais suscetíveis a defeitos? 4) As ferramentas utilizadas criam uma lista dos defeitos identificados?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Detectar e facilitar a identificação de defeitos diretamente no código-fonte. 2) Reportar à equipe informações a cerca dos possíveis defeitos no código-fonte e suas soluções. 3) Identificar e facilitar o entendimento em relação aos trechos do código-fonte mais suscetíveis a apresentarem defeito. 4) Facilitar a identificação dos defeitos mais críticos, probabilidades de ocorrência e seus efeitos a fim de manter um guia com boas práticas.	S/N S/N S/N
12	Ferramenta de análise dinâmica de código	1) São utilizadas ferramentas para analisar dinamicamente a aplicação? 2) Tais ferramentas reportam as possíveis vulnerabilidades? 3) Identificam quais funcionalidades são mais suscetíveis a vulnerabilidades? 4) As ferramentas utilizadas criam uma lista das vulnerabilidades identificadas?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Facilitar a identificação de defeitos no comportamento da aplicação. 2) Reportar à equipe informações a cerca dos possíveis defeitos no comportamento da aplicação e suas soluções. 3) Identificar e facilitar o entendimento em relação às funcionalidades mais suscetíveis a apresentarem defeito. 4) Facilitar a identificação dos defeitos mais críticos, probabilidades de ocorrência e seus efeitos.	S/N S/N S/N
13	Ferramenta de revisão de código	1) São utilizadas ferramentas para analisar estaticamente o código-fonte? 2) Tais ferramentas reportam as possíveis vulnerabilidades? 3) Identificam quais trechos de código são mais suscetíveis a vulnerabilidades? 4) As ferramentas utilizadas criam uma lista das vulnerabilidades identificadas?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Facilitar a identificação de defeitos diretamente no código-fonte. 2) Reportar à equipe informações a cerca dos possíveis defeitos no código-fonte e suas soluções. 3) Identificar e facilitar o entendimento em relação aos trechos do código-fonte mais suscetíveis a apresentarem defeito. 4) Facilitar a identificação dos defeitos mais críticos, probabilidades de ocorrência e seus efeitos.	S/N S/N S/N

14	Avaliação de vulnerabilidades	1) As vulnerabilidades identificadas são gerenciadas? 2) Tais vulnerabilidades são classificadas e organizadas por prioridades? 3) As listas de vulnerabilidades são atualizadas e reclassificadas? 4) Para cada vulnerabilidade são identificadas as ferramentas para detecção destas?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Fornecer informações de segurança, atualizar e educar a equipe em relação as consequências das vulnerabilidades mais comuns encontradas em aplicações web. 2) Atualizar lista de vulnerabilidades atualizadas, suas probabilidades de ocorrência e seus efeitos. 3) Manter lista de contramedidas para as vulnerabilidades identificadas e manter um guia atualizado para a equipe. 4) Utilizar ferramentas adequadas para cada tipo de vulnerabilidades identificadas a fim de ter efetividade no procedimento.	S/N S/N S/N S/N
15	Teste de penetração	1) São utilizadas ferramentas para teste de penetração? 2) Tais ferramentas identificam as possíveis vulnerabilidades? 3) Tais ferramentas realizam as tentativas de acesso a aplicação? 4) As ferramentas utilizadas detalham as vulnerabilidades identificadas?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Determinar como um usuário mal-intencionado pode obter acesso não autorizado aos ativos da organização. 2) Identifica as vulnerabilidades para determinar se é possível realizar o acesso não autorizado no sistema. 3) Explorar as vulnerabilidades para determinar se é possível realizar o acesso não autorizado no sistema. 4)Manter uma lista com as vulnerabilidades identificadas no teste de penetração.	S/N S/N S/N S/N
16	*Fuzz testing*	1) São utilizadas ferramentas para *Fuzz Testing*? 2) Tais ferramentas geram as entradas aleatórias e inválidas necessárias? 3) Tais ferramentas realmente exploram as vulnerabilidades? 4) As ferramentas utilizadas detalham as vulnerabilidades identificadas?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Observar o comportamento após geração de entradas aleatórias ou inesperadas. 2) Observar os possíveis defeitos, como lançamento de exceções imprevistas, colapso do cliente ou servidor, entre outros. 3) Analisar os efeitos de uma falha ao inserir entradas inválidas ou aleatórias, o que constatará que o teste foi bem sucedido. 4) Utilizar informações obtidas para desenvolver contramedidas de segurança.	S/N S/N S/N S/N
17	Teste baseado em risco	1) São realizados testes baseados em risco? 2) Os testes foram baseados na documentação da modelagem de riscos? 3) Os testes exploram cada risco? 4) Os riscos são acompanhados e controlados?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Identificar os aspectos que podem prejudicar o funcionamento correto da aplicação. 2) Facilitar e agilizar os testes baseados em riscos. 3) Executar os testes a partir dos requisitos de segurança e modelagem de riscos. 4) Gerenciar a probabilidade e impacto de cada risco, bem como testá-los e priorizá-los.	S/N S/N S/N S/N

18	Revisão de segurança de código	1) São realizadas revisões de segurança de código? 2) Tais revisões reportam as possíveis vulnerabilidades? 3) Identificam quais trechos de código são mais suscetíveis a vulnerabilidades? 4) A revisão de segurança lista as vulnerabilidades identificadas e suas mitigações?	0 até 1 0 até 1 0 até 1 0 até 1	0% até 100%	1) Detectar e facilitar a identificação de defeitos relacionados a segurança diretamente no código-fonte. 2) Reportar à equipe informações os possíveis defeitos relacionados a segurança no código-fonte e suas soluções. 3) Identificar e facilitar o entendimento em relação aos trechos do código-fonte mais suscetíveis a defeito de segurança. 4) Facilitar a identificação dos defeitos de segurança mais críticos, ocorrência e seus efeitos.	S/N S/N S/N S/N

A.3.1 Seção E: Perguntas gerais sobre a implantação

As perguntas a seguir foram aplicadas nas entrevistas com os envolvidos por parte da equipe. O objetivo dessas perguntas é levantar informações em relação a implantação das técnicas e ferramentas para detecção de vulnerabilidades nas atividades de desenvolvimento ágil de software realizadas nas organizações estudadas.

20) Qual a motivação para a implantação de segurança no desenvolvimento ágil de software?

21) Porque foram escolhidas essas técnicas e ferramenta de detecção de vulnerabilidades?

22) Como foi realizada a implantação dessas técnicas e ferramentas?

23) Qual foi a estratégia utilizada para a implantação dessas técnicas e ferramenta de detecção de vulnerabilidades?

24) Quais foram as dificuldades e limitações encontradas na implantação das técnicas e ferramentas?

25) Quais foram as lições aprendidas com a implantação das técnicas e ferramentas?

26) Comentários e *Feedback:* É possível enviar para mail@mail.com ou por aqui:

OBS: A Tabela 5 apresenta o as técnicas e ferramenta de detecção de vulnerabilidades mais relevantes na literatura, bem como seus benefícios.

Table 5: Resumo dos benefícios das técnicas e ferramentas de segurança encontradas na literatura

Técnicas e Ferramentas	Benefícios
Especialista em segurança	Especificar, aplicar e acompanhar critérios e políticas de segurança durante o projeto.
Requisitos de segurança	Identificar os objetivos e restrições de segurança na aplicação.
Abuse cases/stories	Especificar a interação entre atores e o sistema, no qual seus resultados são prejudiciais ao sistema.
Misuse cases/stories	Descrever uma função que o sistema não deve permitir.
Modelagem de ameaças	Orientar e auxiliar a equipe na identificação das necessidades de recursos de segurança nas áreas mais críticas da aplicação.
Técnicas de contramedidas de segurança	Introduzir proteção contra uma ameaça. Uma vulnerabilidade cuja exposição ao risco pode ser atenuada com a implementação de uma contramedida, mitigação de segurança.
Análise de riscos	Identificar as vulnerabilidades existentes, probabilidade de ataques e seus impactos.
Gestão de respostas a incidentes	Auxiliar na resposta eficiente em caso de uma violação da política de segurança.
Design de segurança	Definir a estrutura geral da aplicação, tendo como ponto de vista a segurança e identificar os componentes cujo funcionamento correto é essencial para a segurança.
Codificação segura	Padronizar a codificação, utilizando práticas como: uso de compiladores mais recentes, organização de listas de funções proibidas ou inseguras, revisão de código e boas práticas de programação.
Ferramenta de análise estática de código	Analisar o código-fonte da aplicação Verificar se os mecanismos de proteção construídos para o sistema irão proteger o sistema de acessos indevidos.
Ferramenta de análise dinâmica de código	Verificar se os mecanismos de proteção construídos para o sistema irão proteger o sistema de acessos indevidos. Assim, o testador utiliza uma ferramenta especificas e simula o papel do indivíduo que deseja acessar o sistema.
Ferramenta de revisão de código	Detectar com eficiência inconsistências no código-fonte, utilizando ferramentas específicas.
Avaliação de vulnerabilidades	Fornecer informações de segurança, atualizar, educar desenvolvedores, designers, arquitetos e organizações a respeito das consequências das vulnerabilidades mais comuns encontradas em aplicações web.
Teste de penetração	Determinar como um usuário mal-intencionado pode obter acesso não autorizado aos ativos da organização.
Fuzz testing	Fornecer entradas inválidas, inesperadas ou aleatórias em um sistema e observa os possíveis defeitos, É mais fácil de implementar pois possui um design para testes mais simples e atribui entradas aleatórias para o sistema.
Teste baseado em risco	Identificar fatores de risco associados aos requisitos do software, sua priorização, probabilidade de ocorrência e impacto, elaborando assim testes relacionados a estes.
Revisão de segurança de código	Detectar com eficiência inconsistências relacionadas a segurança no código-fonte, utilizando ferramentas específicas.

Table 6: Exemplos de técnicas e ferramentas para detecção de vulnerabilidades

Técnicas e Ferramentas	Exemplos
Requisitos de segurança	UMLSec, *Abuse cases/stories, Misuse cases/stories*, etc.
Modelagem de ameaças/risco	Árvore de ameaças, STRIDE, DREAD, etc.
Técnicas de contramedidas de segurança	Politicas de segurança, lista de vulnerabilidades e ameaças, BSIMM, etc.
Análise de riscos	OCTAVE, STRIDE, DREAD, etc.
Abuse/misuse case/stories	UMLSec, Secure UML, UML, etc.
Ferramenta de modelagem de ameaça	AS/NZS 4360:2004, STRIDE, DREAD, etc.
Revisão de código	políticas de segurança, guias de boas práticas, ferraentas específicas, etc.
Gestão de respostas a incidentes	Microsoft Security Response Center (MSRC). Computer Security Incident Handling Guide - NIST. SANS Institute -The Incident Handlers Handbook.
Design de segurança	UMLSec, Secure UML, Árvore de ameaças, etc.
Codificação segura	Revisão de código, bibliotecas seguras, programação em pares, etc.
Análise estática de código	*Parsing*, Análise de fluxo de dados, Análise de fluxo de controle, etc.
Análise dinâmica de código	Teste de penetração, *Fuzz testing*, Injeção de falha, etc.
Ferramenta de revisão de código	FindBugs, FindSecBugs, Codacy, etc.
Ferramenta de análise estática de código	PreFast, RIPS, Veracode Static Analysis, etc.
Ferramenta de análise dinâmica de código	FormatGuard, StackShield, Valgrind, etc.
Avaliação de vulnerabilidades	Injeções SQL, Estouro de memória, Força Bruta, *Fuzz testing*, etc.
Teste de penetração	Netcraf, Httprint, WebRecon, etc.
Fuzz testing	OWASP WSFuzzer, wFUZER
Teste baseado em risco	Inside-Out, Outside-in, Taxonomy Based questionnaire –TBQ, etc.
Revisão de segurança de código	Kiuwan, Seeker, Codacy, etc.