

**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:** O objetivo geral deste estudo consiste em identificar, descrever e comparar as técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas em três organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis.

Para atingir o objetivo principal, será elaborado um instrumento com base na revisão bibliográfica que identifique e descreva as técnicas e ferramentas de detecção de vulnerabilidades aplicadas nas organizações que utilizem métodos ágeis, quais técnicas e ferramentas são aplicadas pelas organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis; quais as características da aplicação de técnicas e ferramentas para detecção de vulnerabilidades e desenvolvimento ágil de software, como: razões, dificuldades, limitações e lições aprendidas nas organizações que utilizam métodos ágeis; necessidades de treinamento das organizações em termos de técnicas e ferramentas para detecção de vulnerabilidades em desenvolvimento ágil de software; técnicas e ferramentas para detecção de vulnerabilidades em desenvolvimento ágil de software aplicados nas organizações brasileiras em relação aos previstos pela literatura; e compare os estados da prática dessas técnicas e ferramentas entre as organizações.

Este projeto de pesquisa utiliza um estudo realizado na Noruega com organizações desenvolvedoras de software que utilizam métodos ágeis, o qual lista técnicas e ferramentas comuns a três processos de segurança: OWASP CLASP, MS SDL e Touchpoints e compara o estado da prática em relação a adoção dessas técnicas e ferramentas, competências e formação da equipe em segurança de software.

Assim, serão identificadas e descritas as técnicas e ferramentas de segurança que são comuns nos seguintes processos: OWASP CLASP, de McGraw e as atividades propostas por Howard e Lipner.

Convidaremos 3 organizações desenvolvedoras de software que utilizam métodos ágeis para a participação no experimento. Tais organizações estão sendo definidas, bem como segmento e nº de participantes. Será apresentada uma lista com técnicas e ferramentas de segurança comum aos processos OWASP CLASP, de McGraw e Howard e Lipner, as quais contribuem para a aplicação de segurança nas diferentes fases do desenvolvimento de software e consequentemente detectar possíveis vulnerabilidades na aplicação.

Entende-se por vulnerabilidade a característica de algo que está sujeito a críticas por apresentar falhas ou incoerências; fragilidade. Dessa forma, será comparado o estado da prática em relação a utilização de técnicas e ferramentas de segurança, aplicadas durante o desenvolvimento de software utilizando métodos

ágeis entre as três organizações, como: a adoção de técnicas, competências formação e habilidades da equipe em segurança de software, compreensão das práticas de segurança utilizadas, recursos existentes e as necessidades de formação da equipe para auxiliar a gestão a tomar decisões corretas em relação às práticas de segurança.

Antes das entrevistas, os participantes do experimento serão convidados pessoalmente. Caso elas aceitem, retornaremos o contato com todas as informações necessárias para realizá-lo. O experimento ocorrerá através de um questionário que será disponibilizado no Google Forms, bem como entrevistas por Skype caso não haja disponibilidade destas serem presenciais.

**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 da Engenharia de Software convencional. Um dos principais objetivos dos métodos ágeis é a interação com o cliente, entregas incrementais, flexibilidade para mudanças e foco principal no desenvolvimento do software. O desenvolvimento ágil oferece benefícios importantes, como responder rapidamente às mudanças de mercado, de negócio e tecnológicas. 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.

O estudo exploratório e qualitativo proporcionou o estudo de caso em três organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis.

Tal estudo parte da visão da organização que aplica as técnicas e ferramentas para detecção de vulnerabilidades no desenvolvimento ágil de software. Esta pesquisa visa identificar e descrever as técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas em três organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis e comparar as técnicas e ferramentas aplicadas, seus níveis de aplicação, capacidade e necessidade de formação da equipe.

A coleta de dados será realizada por meio de entrevistas e técnicas de análise de conteúdo nos dados coletados. Utiliza-se como base para as questões relacionadas a habilidades e necessidades de formação, um estudo realizado na Noruega onde foram identificadas as técnicas e ferramentas em três processos de segurança em desenvolvimento de software, descritas e comparadas as habilidades e necessidades de formação da equipe em segurança de software em duas organizações desenvolvedoras de software que aplicam métodos ágeis.

A partir dos resultados dessa pesquisa, espera-se identificar as dificuldades e oportunidades para conciliar métodos ágeis e segurança de software, nível de aplicação, habilidades e necessidades de formação da equipe em segurança de software nas três organizações estudadas.

**Introdução:** 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. 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). Para sistemas maiores ou críticos, é necessário, porém uma abordagem mais detalhada e rígida, conhecida também como orientada a plano, tradicional (SOMMERVILLE, 2011).

No entanto, em ambientes com constantes mudanças, uma abordagem ágil é recomendável. Os métodos ágeis foram criados por um grupo de 17 especialistas em desenvolvimento de software e tiveram seu ápice no ano 2000. Pela 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, 2010b).

Uma das definições de vulnerabilidade encontradas no dicionário Michaelis (2016b) é que esta é a característica de algo que apresenta falhas ou incoerências; fragilidade.

De acordo com a OWASP (2010b) 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 (2006) apontam que práticas de segurança incorporadas no ciclo de vida de desenvolvimento de software diminuem a probabilidade de existirem vulnerabilidades no sistema.

**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:

Processos	Descrição
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. É um conjunto de atividades baseado em função e guiado pelas melhores práticas de segurança formalizadas.
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, garantia de segurança, misuse cases, análise estática e dinâmica de código, teste de penetração 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	É um processo que incorpora atividades de segurança em um modelo de ciclo de vida de desenvolvimento de software cascata. Uma modelagem de riscos deve ser realizada na fase de especificação de requisitos constando as possíveis ameaças e vulnerabilidades. Após a modelagem de ameaças uma política de segurança deverá ser definida para que fique claro como a segurança será alcançada.
TSP Secure	Processo para Equipes de Desenvolvimento de Software Seguro é um processo que tem por objetivo orientar as equipes no desenvolvendo de produtos de software, auxiliando as organizações a estabelecerem uma prática de engenharia madura e disciplinada, que produz rapidamente software seguro, confiável.
Howard e Lipner	Incorporam atividades de segurança aos métodos ágeis. É necessário ter atenção, pois algumas atividades podem levar a um processo não ágil.

Figure 1: Processos de desenvolvimento de software seguro selecionados no estudo

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. As técnicas e ferramentas serão classificadas em: concepção, modelagem de ameaças e gerenciamento de riscos, *design* seguro e codificação, ferramentas de segurança, testes de segurança e distribuição.

Concepção	<i>Design</i> e codificação segura
Existe um especialista em segurança	<i>Design</i> de segurança
Estabelece requisitos de segurança	Codificação segura
Escreve <i>abuse/misuse cases/stories</i>	
Modelagem de ameaças e gestão de riscos	Ferramentas de Segurança
Modelagem de ameaças	Ferramenta de modelagem de ameaça/risco
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 resposta a incidentes	Avaliação de vulnerabilidades
	Teste de penetração
	<i>Fuzz testing</i>
	Teste baseado em risco
	Revisão de segurança de código

Figure 2: Técnicas e ferramentas para detecção de vulnerabilidades mais citadas nos processos

**Metodologia proposta:** Uma pesquisa exploratória e qualitativa, composta por entrevistas, aplicação de questionários e análise de conteúdo. Para questões relacionadas a habilidades e necessidade de formação da equipe, 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. Em consideração a isso, haverá uma distinção entre as principais atividades de segurança e as atividades que podem ser aproveitadas para proporcionar segurança.

A pesquisa exploratória e qualitativa proporcionará o estudo de caso em três organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis. Tal estudo parte da visão da organização que aplica as técnicas e ferramentas para detecção de vulnerabilidades no desenvolvimento ágil de software.

Esta pesquisa visa identificar e descrever as técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas em três organizações brasileiras desenvolvedoras de software que utilizam métodos ágeis e comparar as técnicas e ferramentas aplicadas, seus níveis de aplicação, capacidade e necessidade de formação da equipe.

Para a aplicação dos estudos de casos foi desenvolvido um instrumento com perguntas relacionadas a habilidades e necessidades de treinamento, as quais foram baseadas em um estudo de caso realizado na Noruega com três organizações desenvolvedoras de software que utilizam métodos ágeis, bem como na revisão bibliográfica realizada para identificar e descrever os principais processos, suas técnicas e ferramentas. Para o desenvolvimento do instrumento da pesquisa a seguinte estratégia foi realizada:

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 selecionadas as técnicas e ferramentas de segurança comumente

aplicadas pelos seguintes processos: OWASP CLASP, Howard e Lipner e de McGraw, a fim de manter a conformidade com o estudo realizado na Noruega, comparar e listar os benefícios que foram relatados na literatura para essas técnicas e ferramentas.

A coleta de dados será realizada por meio de entrevistas e técnicas de análise de conteúdo nos dados coletados. Utiliza-se como base para as questões relacionadas a habilidades e necessidades de formação, um estudo realizado na Noruega onde foram identificadas as técnicas e ferramentas em três processos de segurança em desenvolvimento de software, descritas e comparadas as habilidades e necessidades de formação da equipe em segurança de software em duas organizações desenvolvedoras de software que aplicam métodos ágeis.

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

De forma mais detalhada, a estratégia de estudo utilizada nesta pesquisa é o estudo de caso múltiplo, que tem por intuito avaliar e comparar as técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas pelas organizações e comparar o comportamento e resultado entre elas. Baseado nesta estratégia de múltiplos casos, os estudos desta pesquisa foram 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 que”, as quais tem a função de esclarecer a natureza do estudo. Estas, devem estar em conformidade com os objetivos e questões dessa pesquisa que analisou as técnicas e ferramentas de detecção de vulnerabilidades em desenvolvimento ágil de software aplicadas pelas organizações brasileiras, seus benefícios e o porquê da incorporação de determinada técnica ou ferramenta.

**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 problema do “caso” do estudo de caso. Nesta pesquisa, o “caso” seriam 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.

**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.

É recomendável que um projeto de pesquisa seja estruturado por meio desses cinco componentes, a fim de auxiliar na coleta, análise e conclusão dos resultados obtidos. Neste projeto de pesquisa e em conjunto com a metodologia de múltiplos casos propostas por Yin (2013).

**Riscos:** Garantir o sigilo dos participantes, confiabilidade, validade de conteúdo, validade de construção, validade externa.

**Benefícios:** Como resultado para a pesquisa se espera 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, se espera 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 utilização destas.

Com objetivo de obter esses resultados, uma comparação do estado da prática do uso de técnicas e ferramentas entre as organizações que utilizam métodos ágeis no desenvolvimento de software foi realizada por meio de um instrumento desenvolvido, composto pelas técnicas e ferramentas de detecção de vulnerabilidades no desenvolvimento ágil de software relatadas na literatura e mais utilizadas nos processos escolhidos, seus benefícios, dificuldades, limitações e motivação da utilização destas.

O intuito de comparar o estado da prática do uso de técnicas e ferramentas de detecção de vulnerabilidades no desenvolvimento ágil de software entre as organizações consiste em avaliar a adoção de técnicas e ferramentas de segurança, limitações e dificuldades na adoção, benefícios e motivação e se estas são adequadas para o contexto de organizações brasileiras desenvolvedoras de software que aplicam métodos ágeis.

**Desfecho Primário:** Este projeto visa analisar os processos, atividades e ferramentas de segurança, habilidades da equipe e necessidades de formação de três organizações que utilizam desenvolvimento ágil no Brasil.

Organizações que aplicam métodos ágeis utilizam atividades que melhor se adequam às suas necessidades de processo e de negócios. Assim, entender as semelhanças, diferenças, benefícios, dificuldades e motivação entre as práticas de segurança das organizações pode auxiliar a replicação e adoção de técnicas e ferramentas de segurança de software entre diferentes organizações.

**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

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. Citado na página 31.

MICHAELIS. Dicionário brasileiro da língua portuguesa: Segurança. 2016. Disponível em: <http://michaelis.uol.com.br/busca?r=0&f=0&t=0&palavra=seguran%C3%A7a>.

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

OWASP. CLASP Concepts. 2010. Disponível em: [https://www.owasp.org/index.php/CLASP\\_Concepts](https://www.owasp.org/index.php/CLASP_Concepts). Citado 2 vezes nas páginas 50 e 51.

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. Availability, reliability, and security in information systems. SINTEF. Availability, Reliability, and Security in Information Systems., Conference Publishing Services (CPS), p. 548–555, 2016.

SOMMERVILLE, I. Software Engineering. 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 através da consciência do aprendizado contínuo.

O Survey está dividido em cinco seções: Seção A aborda informações gerais sobre a organização. A Seção B foca em informações gerais sobre os projetos. Já a Seção C está relacionado a aptidões, interesses e uso atual de técnicas e ferramentas de segurança de software, a Seção D está focada em 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 Development (FDD)
- Lean Software Development
- Crystal Methods
- Kanban
- Agile Unified Process (AUP)
- Dynamic Systems Development Method (DSDM)
- Outro (especificar)

8) Você possui experiência em segurança de software?

- Sim
- Não

9) N° 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 3: 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.

## **Processos e benefícios da governança de dados**

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%. Adicionalmente cada pergunta tem peso de 0,25 na porcentagem total de implantação de um processo, portanto a porcentagem de implantação de uma técnica ou ferramenta é calculada da seguinte forma:

$\%IMP \text{ Processo} = Soma ((P1 * 0,25) + (P2 * 0,25) + (P2 * 0,25) + (P4 * 0,25))$   
P1 até P4 representa o valor de implantação da pergunta que varia de 0 até 1 (0% até 100%).

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

### **A.3 Instrumento do estudo de caso**

Table 1: Instrumento do estudo de caso.

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

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

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

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

Table 2: 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.
<i>Abuse cases/stories</i>	Especificar a interação entre atores e o sistema, no qual seus resultados são prejudiciais ao sistema.
<i>Misuse cases/stories</i>	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.
<i>Design</i> 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 específicas 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.
<i>Fuzz testing</i>	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 3: Exemplos de técnicas e ferramentas para detecção de vulnerabilidades

Técnicas e Ferramentas	Exemplos
Requisitos de segurança	UMLSec, <i>Abuse cases/stories</i> , <i>Misuse cases/stories</i> , etc.
Modelagem de ameaças/risco	Árvore de ameaças, STRIDE, DREAD, etc.
Técnicas de contramedidas de segurança	Políticas de segurança, lista de vulnerabilidades e ameaças, BSIMM, etc.
Análise de riscos	OCTAVE, STRIDE, DREAD, etc.
<i>Abuse/misuse case/stories</i>	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, ferramentas específicas, etc.
Gestão de respostas a incidentes	Microsoft Security Response Center (MSRC).
	Computer Security Incident Handling Guide - NIST.
	SANS Institute -The Incident Handlers Handbook.
<i>Design</i> de segurança	UMLSec, Secure UML, Árvore de ameaças, etc.
Codificação segura	Revisão de código, bibliotecas seguras, programação em pares, etc.
Análise estática de código	<i>Parsing</i> , Análise de fluxo de dados, Análise de fluxo de controle, etc.
Análise dinâmica de código	Teste de penetração, <i>Fuzz testing</i> , 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, <i>Fuzz testing</i> , etc.
Teste de penetração	Netcraf, Httpprint, WebRecon, etc.
<i>Fuzz testing</i>	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.