Explorando a especificação de cenários de teste de segurança no desenvolvimento orientado por comportamento (Behavior Driven Development)

Maria Yêda Lima, Carla Silva, and Renata Souza

Centro de Informática, Universidade Federal de Pernambuco, Recife, Brasil
{myml,ctlls,rmcrs}@cin.ufpe.br

Resumo Contexto: O aumento da coleta de dados pessoais tem levantado preocupações também crescentes sobre a segurança da informação em sistemas de software dos quais a sociedade está cada vez mais dependente. Neste cenário, desenvolver software seguro que esteja protegido de ameaças é uma preocupação que deve ser considerada na Engenharia de Requisitos. O desenvolvimento orientado por comportamento (Behavior Driven Development - BDD) é uma abordagem ágil de desenvolvimento de software que tem a característica de estimular a contínua interação entre o time de desenvolvimento e o cliente, facilitando a comunicação entre as partes, além de integrar a especificação de cenários de teste com a especificação dos requisitos do sistema. **Objetivo:** O objetivo geral desta pesquisa é investigar como a indústria de desenvolvimento de software vem realizando a especificação de cenários de teste em projetos que adotam o BDD e descrever um conjunto de cenários de teste de segurança que poderão ser usados por aqueles que desejam aprender sobre segurança de sistemas, bem como por engenheiros de software que desejam desenvolver sistemas mais seguros por projeto (secure by design). **Metodologia:** Esta pesquisa propõe inicialmente a realização de estudo da literatura para delimitar o problema de pesquisa a ser resolvido e analisar os trabalhos relacionados. Em seguida, um survey será conduzido para investigar as práticas que a indústria adota e as dificuldades que a indústria enfrenta em projetos que adotam BDD. Também realizaremos um estudo exploratório baseado em entrevista em empresas da indústria nacional que adotam o BDD, a fim de identificar boas práticas já adotadas e oportunidades de melhorias no processo de especificação de cenários de teste de segurança da informação. Com base nestes dados coletados, vamos definir cenários de teste de segurança que atendam as necessidades identificadas na literatura e na indústria. Por fim, esses cenários de teste de segurança serão avaliados considerando a sua utilidade e seu impacto por meio de um experimento controlado. **Resultados:** Com os resultados obtidos nesta pesquisa, pretendemos contribuir com a definição de artefatos de teste de segurança que auxiliem no desenvolvimento de software mais seguro.

Keywords: Estudo exploratório · Pesquisa Qualitativa · Engenharia de Requisitos · Secure by Design · Desenvolvimento Orientado por Comportamento · Behavior Driven Development

1 Caracterização do Problema

De acordo com o Departamento de Segurança Interna dos Estados Unidos, aproximadamente 90% das vulnerabilidades de software podem ser rastreadas no projeto do software e nas fraquezas da codificação [9]. Contudo, existe uma falta de conhecimento sobre o ciclo de vida de Engenharia de Software e em relação a como as vulnerabilidades são introduzidas e/ou removidas pelos desenvolvedores [13]. Neste cenário, desenvolver software seguro é uma preocupação que deve ser considerada nas atividades do processo de Engenharia de Requisitos. [5] comentam que para desenvolver sistemas e serviços baseados em software seguros é essencial testar o software para detectar falhas, bem como vulnerabilidades de segurança. Nesse contexto, surgem os testes de segurança ou de penetração, que tem o propósito de identificar vulnerabilidades antes que elas sejam exploradas. [10] comentam que se as vulnerabilidades de software não forem tratadas adequadamente podem resultar em sérias consequências negativas tanto para usuários como para provedores dos serviços baseados em software e que essas questões devem ser abordadas nos estágios iniciais de desenvolvimento de software, aumentando a conscientização dos engenheiros de software sobre o desenvolvimento de software seguro [17] [10]. [23] destacam que fatores organizacionais e recursos humanos influenciam o sucesso das iniciativas de desenvolvimento de software seguro e que o treinamento de desenvolvedores pode motivar as equipes de desenvolvimento a adotar práticas de segurança. E acrescenta destacando que o trabalho de segurança diz respeito aos desenvolvedores que precisam integrar ferramentas e atividades em seu trabalho diário, mas também da organização como um todo, que deve orientar e apoiar os esforços de segurança estabelecendo segurança como um requisito explícito dentro das equipes de desenvolvimento ágil. O desenvolvimento orientado por comportamento (Behavior Driven Development - BDD) é uma abordagem ágil de desenvolvimento de software que tem a característica de estimular a contínua interação entre o time de desenvolvimento e o cliente, facilitando a comunicação entre as partes, além de integrar a especificação de cenários de teste com a especificação dos requisitos do sistema.

Considerando esse cenário, o objetivo geral desta pesquisa é investigar como a indústria de desenvolvimento de software vem realizando a especificação de cenários de vulnerabilidade em projetos que adotam o BDD e definir um conjunto de artefatos de teste de segurança que ajudem aqueles que desejam aprender como desenvolver sistemas de software mais seguros.

2 Fundamentação Teórica

2.1 Segurança Cibernética

Segurança é uma medida da capacidade do sistema de proteger dados e informações de acesso não autorizados enquanto ainda fornece acesso a pessoas e sistemas autorizados [3]. A segurança no contexto de sistemas de softwrare possui cinco características, que são, *confidencialidade* - protege dados ou serviços

contra acesso não autorizado; *integridade* - previne a manipulação não autorizada de dados ou serviços; *disponibilidade* - garante a disponibilidade do sistema para uso legítimo; *autenticação* - identifica os atores envolvidos em uma transação e verifica se eles são quem afirma ser; *não repúdio* - certifica de que o garante que o remetente de uma mensagem não pode negar posteriormente ter enviado a mensagem, e que o destinatário não pode negar ter recebido a mensagem [3][22]. A segurança cibernética pode ser definida como o conjunto de ferramentas, técnicas, políticas, medidas de segurança, estratégias de mitigação de riscos, ações, treinamento, boas práticas, garantia de segurança e por sua importância global envolve a proteção de informações por meio da detecção, prevenção e resposta a ataques cibernéticos [12]. No universo da segurança cibernética existem algumas terminologias, como o *ciberespaço* - que é um domínio global dentro do mundo da informação; *vulnerabilidades* - são as falhas em um sistema ou no design que permitem que um invasor execute comandos maliciosos, acesse dados de forma não autorizada e/ou realize vários ataques de negação de serviço; *ameaças* - são ações tomadas para obter um benefício de falhas de segurança em um sistema e impactá-lo negativamente; *ataques* - são as ações tomadas para danificar um sistema ou perturbar suas operações de rotina explorando vulnerabilidades usando várias ferramentas e técnicas, sendo geralmente motivados por recompensa financeira. Além disso, existem algumas vulnerabilidades mais comuns na segurança cibernética : *negação de serviço (DoS)*, *malware*, *phishing*, *injeção de SQL*, *sequestro de sessão e ataques do tipo Man-in-the-Middle*. Os avanços tecnológicos possibilitaram altos investimentos em TI gerando lucro e prosperidade. Contudo, as violações de dados, ataques cibernéticos e violações de privacidade tornaram-se cada vez mais frequentes [1], comprometendo informações pessoais dos usuários, tais como nome, endereço, números de cartão de crédito e informações médicas [24].

2.2 Behavior-Driven Development

As metodologias ágeis de desenvolvimento de software surgiram da necessidade de tornar o processo de desenvolvimento menos complexo, buscando mitigar as limitações dos métodos tradicionais de desenvolvimento de software. O desenvolvimento orientado por comportamento (BDD) é uma das abordagens que o desenvolvimento ágil de software utiliza e que tem como uma das características estimular a interação entre as partes interessadas no processo de desenvolvimento de software [2]. Além disso, o BDD permite a adoção de um vocabulário comum na organização promovendo o compartilhamento, coordenação e colaboração do conhecimento. [14] destacam que embora as práticas de BDD não tenham sido desenvolvidas para projetos de grande escala, ainda assim, tem potencial para oferecer benefícios contribuindo de maneira positiva para lidar com os desafios de colaboração, comunicação, requisitos, verificação de projetos de grande escala, entre outros. O estudo de [4] relatam que os desafios mais críticos associados ao BDD são de colaboração, falta de treinamento, manutenção e um modelo para escrever especificações. E os benefícios relatados pelos participantes desse estudo

foram: melhor capacidade de testar requisitos, documentação aprimorada, melhor captura do conhecimento do domínio, melhor compreensão e implementação do software.

3 Contribuições Esperadas

Essa tese se preocupa em elaborar cenários de teste de segurança que atendam as necessidades identificadas na literatura e na indústria. Além disso, propor um guia para auxiliar o desenvolvimento de software e também no treinamento de novatos na área de desenvolvimento seguro de software.

Os objetivos específicos desta tese são: (i) Apresentar o estado atual das práticas de BDD que a indústria adota e as dificuldades que a indústria enfrenta na especificação de requisitos e testes em projetos que usam BDD; (ii) Apresentar os resultados do estudo exploratório baseado em entrevista e validado em empresas da indústria nacional que adotam o BDD; (iii) Realizar um experimento controlado a fim de levantar quais são as dificuldades, desafios e facilidades dos profissonais no processo de especificação BDD no contexto de segurança cibernética; (iv) Elaborar um conjunto de artefatos BDD focados em cenários de vulnerabilidades que auxiliem tanto aqueles que desejarem adquirir conhecimento na área de segurança cibernética, bem como os engenheiros de software no desenvolvimento de software mais seguros.

4 Método de Pesquisa

Na condução desta pesquisa, realizaremos 4 estudos formalmente estruturados.

Estudo Bibliográfico. Realizamos um estudo bibliográfico sobre "qualidade da especificação de testes em projetos que usam BDD"e para isso, utilizamos a técnica de snowballing [27] [16] buscando identificar trabalhos sobre o assunto a partir de um conjunto inicial que possuía no seu título e abstract as palavras-chaves "acceptance testing", "quality", "metrics"e "software development"e utilizamos o Google Scholar, ACM, IEEE e SCIENCEDIRECT para realizar as buscas . Definimos 3 (três) critérios de inclusão e 4 (quatro) critérios de exclusão. O processo de pesquisa foi de busca manual e foram selecionados artigos de 2014 à 2020.

Survey. A condução do survey [19] [26] visa entender o estado atual das práticas de BDD que a indústria de software usa para especificar casos de testes.

Entrevistas. Realizaremos um estudo exploratório [28] [18] utilizando o método de entrevistas para a coleta de dados, nas quais serão convidados engenheiros de software que trabalham na indústria nacional. O foco neste estudo será identificar boas práticas já adotadas e oportunidades de melhorias no processo de especificação BDD no contexto de segurança.

Experimento Controlado. O experimento controlado [7] terá como objetivo validar o conjunto de artefatos BDD focados em cenários de vulnerabilidades e terá como foco os engenheiros de software e estudantes que estejam no final da graduação de áreas relacionadas com a nossa pesquisa.

4.1 Ameaças à Validade

As ameaças a validade aqui apresentadas se referem apenas ao survey, pois as Entrevistas e o Experimento Controlado ainda não foram planejados.

Validade Interna. A carta de apresentação do questionário menciona explicitamente o objetivo geral e a importância da pesquisa, garantido o sigilo dos dados, o anonimato da participação e o direito de desistir da pesquisa a qualquer momento. Assim, espera-se que somente o público alvo (desenvolvedores/engenheiros de software) da pesquisa de fato responda o questionário.

Validade Externa. Existe a possibilidade de obtermos um número de participantes pequeno, o que torna impraticável a generalização dos resultados. Para ajudar a aumentar o número de participantes o questionário foi feito em duas versões (português e inglês) e enviado para o máximo de canais disponíveis e de alcance mundial (Twitter, LinkedIn, grupos de pesquisa, listas de pesquisa, etc).

Validade de Conclusão. Pode existir diferença de conhecimento e experiência profissional entre os pesquisadores que analisarão os resultados do survey. Para mitigar esta ameaça, as três pesquisadoras envolvidas no projeto analisarão os dados coletados para a obtenção das conclusões.

Validade de Constructo. O questionário foi elaborado com base em outros questionários aplicados na área de Engenharia de Requisitos e também com base em trabalhos relacionados. Solicitamos que pesquisadores experientes e profissionais da indústria avaliassem o questionário para eliminar qualquer ambiguidade no texto apresentado. O survey será realizado remotamente e os participantes podem desistir a qualquer momento. Para reduzir o risco de desistências, elaboramos um instrumento curto, cujo preenchimento é de aproximadamente 10min.

5 Estado Atual do Trabalho

Realizamos inicialmente uma pesquisa estruturada utilizando a técnica de snowballing com foco na especificação de testes com qualidade voltado para o contexto ágil e analisamos um total de 237 artigos. Contudo, percebemos durante o nosso estudo que nenhum desses trabalhos focavam em cenários de segurança no contexto ágil. Assim, delimitamos o nosso problema de pesquisa na especificação de cenários de vulnerabilidades em projetos que adotam o BDD. Buscando assim, atender tanto o universo do ensino como a indústria.

Estamos planejando um survey para entender o estado atual das práticas de BDD que as empresas na indústria usam para especificar casos de testes. As principais perguntas que queremos obter são :"Como os casos de teste são escritos na prática ?", "Quais são as dificuldades em especificar casos de teste usando as práticas de BDD durante suas atividades ?"e "Como as empresas de desenvolvimento de software contribuem para diminuir as dificuldades que os engenheiros de software enfrentam ao adotar as práticas BDD para especificar casos de teste ?". Confeccionamos um questionário como instrumento de coleta de dados, composto de quatro partes. A primeira e a segunda parte enfocam a caracterização dos entrevistados e a parte organizacional, respectivamente. A terceira e quarta

parte, investigamos as práticas ágeis de BDD na indúntria buscando saber quais as dificuldades e desafios que os profissionais enfrentam para especificar casos de testes. O survey consiste em 16 perguntas, sendo 11 perguntas pré-definidas e 5 perguntas abertas, além das instruções de como conduzir o survey. O survey será validado e aprimorado em duas rodadas: primeiro será revisado e discutido em nosso grupo de pesquisa. Em segundo lugar, iremos rodar um piloto e para isso, iremos solicitar que um especialista em BDD valide o instrumento.

Realizaremos posteriormente um estudo exploratório baseado em entrevistas na indústria nacional que adotam o BDD, a fim de identificar boas práticas já adotadas e oportunidades de melhorias no processo de especificação BDD no contexto de segurança. O nosso foco será entrevistar os engenheiros de software.

Realizaremos posteriormente um experimento controlado para validação do conjunto de artefatos BDD focados em cenários de vulnerabilidades.

6 Trabalhos Correlatos

O estudo realizado por [25] ressaltou a necessidade de usar uma abordagem que desenvolva sistemas de software seguros desde o início e justifica mencionando que estudos revelam que cerca de 50% dos problemas de segurança são o resultado de deficiências de design de software, tais como, incompreensão de requisitos arquiteturalmente significativos, implementação arquitetônica deficiente, violação dos princípios de design no código-fonte e degradação da arquitetura de segurança. O resultado desse estudo foi a elaboração de um catálogo denominado Common Architectural Weakness Enumeration (CAWE) que enumera tipos comuns de vulnerabilidades enraizadas na arquitetura de um software e fornece técnicas de mitigação para abordá-las. O catálogo pode ser utilizado como um guia por profissionais, entre eles, desenvolvedores e designers sobre as fraquezas potenciais no design ou na implementação de sua arquitetura. O CAWE teve como base uma lista existente de tipos conhecidos de vulnerabilidades de software, Common Software Weaknesses Enumeration, que lista todos os possíveis tipos de vulnerabilidades no sistema de software. O CAWE possui 224 deficiências arquitetônicas categorizadas entre 11 táticas de segurança e cada entrada é composta de várias seções, como tática de segurança afetada, tipo de falha, descrição textual, exemplos de código-fonte, técnicas de mitigação, métodos para detectar a falha e fragilidades arquitetônicas.

Em [13] foi desenvolvido um estudo empírico em larga escala para investigar o ciclo de vida de vulnerabilidades de software conhecidas. Buscaram entender como, quando, quais circunstâncias são feitas as contribuições para a introdução de vulnerabilidades em projetos de software, por quanto tempo e como elas ocorrem. E para isso, utilizaram como base o National Vulnerability Database (NVD), que é o repositório de dados padrão de gerenciamento de vulnerabilidades do governo dos EUA, para extrair um conjunto de vulnerabilidades e avaliaram 3.663 vulnerabilidades de 144 categorias em 1.096 projetos diferentes do GITHUB. A conclusão do estudo foi que existe a necessidade de mais pesquisas sobre vulnerabilidades e que ainda não existem evidências empíricas

convincentes sobre as razões pelas quais as vulnerabilidades permanecem em um sistema de software por tanto tempo e nem as possíveis motivações pela quais certas ações são mais propensas a introduzirem vulnerabilidades. Os resultados mostraram que as vulnerabilidades são introduzidas geralmente logo após a criação de novos arquivos, por exemplo, ao aprimorar recursos existentes. Além disso, reforçam a necessicade de investimentos adicionais na área educacional de manutenção e evolução de software, principalmente no que diz respeito em como compreender o código-fonte e diagnosticar seus possíveis problemas quando ocorre uma nova solicitação de alteração. Outro achado do estudo foi que existe uma relação entre aspectos humanos e vulnerabilidades e que a maioria dos problemas de segurança são apresentados por desenvolvedores que apresentam uma alta carga de trabalho. Desenvolvedores experientes e recém-chegados têm aproximadamente as mesmas chances de incorrer em problemas de segurança.

O estudo apresentado em [9] buscou responder se existe a necessidade da educação de codificação segura na indústria e teve como alvo da pesquisa o desenvolvedor de software. O foco da pesquisa foi analisar o fator humano, buscando entender a conscientização e a conformidade dos desenvolvedores de software com as diretrizes de codificação segura. Além disso, investigaram o uso de jogos sérios como ferramenta para aumentar a conscientização de codificação segura dos desenvolvedores de software na indústria. Realizaram uma pesquisa baseada em questionário em larga escala sobre diretrizes de codificação segura com mais de 190 desenvolvedores de software que trabalharam em diferentes indústrias e coletaram os dados em um período de sete meses. Os resultados do estudo indicaram que uma grande quantidade de desenvolvedores de software não estão cientes das diretrizes de codificação segura e que um método para suprir essa falta de conscientização é por meio da educação sobre codificação segura. Também inferiram um conjunto de quinze itens para profissionais da indústria e educadores de segurança cibernética que devem ser levados em consideração juntamente com as diretrizes de codificação segura.

O processo de desenvolvimento de software é complexo que requer um alto nível de interação entre as diversas partes interessadas, pois envolve planejar, desenvolver e testar até que o produto seja liberado. O processo de teste depende da execução de casos de testes que devem ser cuidadosamente especificados, pois descrevem uma condição particular a ser testada objetivando um resultado ou comportamento esperado [15][11]. Contudo, especificar casos de teste é uma atividade que exige um esforço que pode chegar a 40-70% do esforço total no processo de teste. De fato, o esforço é justificado pois as entradas e saídas precisam ser corretamente determinadas para que o requisito possa ser testado da forma mais abrangente possível e a quantidade de casos de teste deve ser pensada de maneira que possa evitar esforços de teste desnecessários [21]. Embora teste de software tenha como as suas principais práticas a detecção de falhas, ele também possibilita a detecção de vulnerabilidades de segurança quando projeta-se sistemas e serviços de software seguros [5]. Os estudos de [6] revelaramm que as empresas de software têm o desafio de aplicar sistematicamente testes de segurança em seus processos, contudo existe uma falta de diretrizes na prática,

bem como estudos empíricos em projetos que adotem testes de segurança ágeis e que a indústria em geral precisa de mais abordagem sistemática da segurança. A falta de conhecimento sobre segurança por equipes ágeis em geral, a grande dependência de testes incidentais e o desconhecimento em testes estáticos para segurança são indicadores de que o teste de segurança é altamente subestimado e que mais esforços devem ser direcionados para eles em equipes ágeis.

O trabalho apresentado em [20] menciona que as práticas de desenvolvimento orientado por comportamento (BDD) são utilizadas para representar testes de aceitação em metodologias ágeis e se faz necessário uma formalização de comportamento de boa qualidade para evitar problemas de requisitos oriundos de documentação ruim, como informações incompletas e requisitos inconsistentes. Porém, existem apenas diretrizes informais para orientar os profissionais na elaboração de seus cenários de BDD e avaliações de qualidade. E buscando resolver essa falta de orientação, propuseram um conjunto de atributos de qualidade e uma lista de verificação baseada em perguntas para auxiliar as avaliações de qualidade dos cenários de BDD. O estudo foi baseado em entrevistas com 18 participantes que compartilharam suas interpretações sobre um conjunto inicial de atributos de qualidade informados na literatura e seus próprios critérios de avaliação pessoal. Consolidaram os resultados em uma única lista com oito atributos recém-redefinidos que serviram como entrada para definir um novo checklist baseado em perguntas. Os autores acreditam que os resultados obtidos podem aprimorar as diretrizes existentes e a capacidade dos profissionais em avaliar a qualidade do cenário BDD, fornecendo-lhes uma diretriz padrão. Contudo, os autores reforçam a necessidade de refinar os achados voltando aos participantes da pesquisa para validar na prática quão úteis são os novos atributos de qualidade e a lista de verificação baseada em perguntas para que possam ser usados em contextos BDD no mundo real e obter resultados confiáveis.

O estudo de [8] buscou identificar os desafios atuais no uso de artefatos de teste para entender por que certos fatores de qualidade são considerados bons ou ruins. Além disso, elaboraram um modelo de qualidade de artefato baseado em atividade que descreve como os artefatos de teste ágeis devem ser. O estudo foi motivado pela mudança comportamental dos profissionais que adotam práticas ágeis ao elaborarem arfetados adicionais a documentação recomendada, gerando uma variedade de artefatos que não são inerentes ao que se propõe o ASD, principalmente artefatos relacionados a garantia da qualidade. Os autores postularam que a qualidade de um artefato de teste depende do stakeholder que o utiliza e das atividades para as quais ele é usado e partindo desse princípio exploraram os fatores de qualidade de artefatos de teste que têm um impacto positivo ou negativo sobre as atividades das partes interessadas. Para entenderem por qual motivo determinado fator de qualidade é considerado bom ou ruim pelos profissionais, estudaram no primeiro momento os desafios atuais no uso de artefatos de teste e mais adiante estenderam as qualidades normativas com uma lista de fatores que descrevem como os artefatos de testes ágeis devem ser. Realizaram uma pesquisa qualitativa na indústria com 18 profissionais de 12 empresas que operavam em sete domínios diferentes. Com os resultados obtidos

detectaram nove desafios e 16 fatores que descrevem a qualidade de seis artefatos de teste na perspectiva dos testadores ágeis e alguns dos desafios estavam relacionados principalmente à linguagem (fraca formulação de requisitos) e falta de rastreabilidade entre os artefatos.

7 Conclusões e trabalhos futuros

A presente pesquisa refere-se a proposta de elaboração de um conjunto de artefatos BDD focados em cenários de segurança que ajudem aqueles que desejam aprender sobre segurança de sistemas, bem como os engenheiros de software a desenvolverem sistemas mais seguros por projeto. Até o presente momento, o Estudo Bibliográfico foi concluído, o Survey está em andamento e a validação do instrumento através da realização de um piloto para refinamento foi concluída. Enviamos o survey para os profissionais da indústria responderem e posteriormente, realizaremos a coleta, codificação e análise dos dados. E as Entrevistas e o Experimento Controlado ainda irão começar.

Referências

1. Agrafiotis, I., Nurse, J.R., Goldsmith, M., Creese, S., Upton, D.: A taxonomy of cyber-harms: Defining the impacts of cyber-attacks and understanding how they propagate. Journal of Cybersecurity **4**(1) (2018)
2. Bahaweres, R.B., Oktaviani, E., Wardhani, L.K., Hermadi, I., Suroso, A., Solihin, I.P., Arkeman, Y.: Behavior-driven development (bdd) cucumber katalon for automation gui testing case cura and swag labs. International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS) pp. 87–92 (2020)
3. Bass, L., Clements, P., Kazman, R.: Software architecture in practice. Addison-Wesley Professional (2003)
4. Binamungu, L.P., Embury, S., Konstantinou, N.: Detecting duplicate examples in behaviour driven development specifications. 2018 IEEE Workshop on Validation, Analysis and Evolution of Software Tests (VST) pp. 6–10 (2018)
5. Ceccato, M., Nguyen, C.D., Appelt, D., Briand, L.C.: Sofia: An automated security oracle for black-box testing of sql-injection vulnerabilities. In: 31st IEEE/ACM Intl. Conference on Automated Software Engineering (ASE). pp. 167–177 (2016)
6. Cruzes, D.S., Felderer, M., Oyetoyan, T.D., Gander, M., Pekaric, I.: How is security testing done in agile teams? a cross-case analysis of four software teams. In: Intl. Conf. on Agile Software Development. pp. 201–216. Springer, Cham (2017)
7. Easterbrook, S., Singer, J., Storey, M.A., Damian, D.: Selecting empirical methods for software engineering research. In: Guide to advanced empirical software engineering, pp. 285–311. Springer (2008)
8. Fischbach, J., Femmer, H., Mendez, D., Fucci, D., Vogelsang, A.: What makes agile test artifacts useful? an activity-based quality model from a practitioners' perspective. In: Proceedings of the 14th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM). ESEM '20, Association for Computing Machinery, New York, NY, USA (2020)
9. Gasiba, T., Lechner, U., Albuquerque, M., Méndez Fernández, D.: Is secure coding education in the industry needed? an investigation through a large scale survey. pp. 241–252 (05 2021). https://doi.org/10.1109/ICSE-SEET52601.2021.00034

10. Gasiba, T.E., Lechner, U., Pinto-Albuquerque, M.: Cybersecurity challenges: Serious games for awareness training in industrial environments. ArXiv **abs/2102.10432** (2021)
11. Graham, D., Veenendaal, E.V., Evans, I., Black, R.: Foundations of Software Testing: ISTQB Certification. Intl Thomson Business Pr (2008)
12. Humayun, M., Niazi, M., Jhanjhi, N., Alshayeb, M., Mahmood, S.: Cyber security threats and vulnerabilities: a systematic mapping study. Arabian Journal for Science and Engineering **45**(4), 3171–3189 (2020)
13. Iannone, E., Guadagni, R., Ferrucci, F., De Lucia, A., Palomba, F.: The secret life of software vulnerabilities: A large-scale empirical study. IEEE Transactions on Software Engineering pp. 1–1 (2022). https://doi.org/10.1109/TSE.2022.3140868
14. Irshad, M., Britto, R., Petersen, K.: Adapting behavior driven development (bdd) for large-scale software systems. Journal of Systems and Software **177**, 110944 (2021)
15. Itkonen, J., Mantyla, M.V., Lassenius, C.: How do testers do it? an exploratory study on manual testing practices. In: Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement. p. 494–497. ESEM '09, IEEE Computer Society, USA (2009)
16. Kitchenham, B., Charters, S.: Guidelines for performing systematic literature reviews in software engineering (2007)
17. Leite, G.S., Albuquerque, A.B.: The importance of safe coding practices and possible impacts on the lack of their application. In: Computer Science On-line Conference. pp. 214–224. Springer (2019)
18. Merriam, S.B., Tisdell, E.J.: Qualitative research: A guide to design and implementation. John Wiley & Sons (2015)
19. Molléri, J.S., Petersen, K., Mendes, E.: An empirically evaluated checklist for surveys in software engineering. CoRR **abs/1901.09850** (2019), http://arxiv.org/abs/1901.09850
20. Oliveira, G., Marczak, S., Moralles, C.: How to evaluate bdd scenarios' quality? In: Proceedings of the XXXIII Brazilian Symposium on Software Engineering (SBES). p. 481–490. Association for Computing Machinery, New York, NY, USA (2019)
21. de Oliveira Neto, F.G., Horkoff, J., Svensson, R., Mattos, D., Knauss, A.:
22. Pedraza-Garcia, G., Astudillo, H., Correal, D.: A methodological approach to apply security tactics in software architecture design. 2014 IEEE Colombian Conference on Communications and Computing (COLCOM) pp. 1–8 (2014)
23. Poller, A., Kocksch, L., Türpe, S., Epp, F.A., Kinder-Kurlanda, K.: Can security become a routine? a study of organizational change in an agile software development group. In: Proc. of the ACM Conf. on Computer Supported Cooperative Work and Social Computing (CSCW). p. 2489–2503. ACM, NY, USA (2017)
24. Romanosky, S.: Examining the costs and causes of cyber incidents. Journal of Cybersecurity **2**(2), 121–135 (2016)
25. da Silva Santos, J.C., Tarrit, K., Mirakhorli, M.: A catalog of security architecture weaknesses. pp. 220–223 (04 2017). https://doi.org/10.1109/ICSAW.2017.25
26. Wagner, S., Fernández, D.M., Felderer, M., Graziotin, D., Kalinowski, M.: Challenges in survey research. CoRR **abs/1908.05899** (2019), http://arxiv.org/abs/1908.05899
27. Wohlin, C.: Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering. Association for Computing Machinery, New York, NY, USA (2014)
28. Yin, R.K.: Estudo de Caso: Planejamento e metodos. Bookman editora (2015)