Quality Assurance como primeira linha de defesa: uma análise da relação entre falhas de software e vulnerabilidades de segurança

Luan Henrique Leite Vilela de Queiroz[1], **André Almeida Silva**[1]

[1]Universidade Federal de Alagoas (UFAL) – Campus Arapiraca

{luan.queiroz, andre.almeida}@arapiraca.ufal.br

Abstract. *The rise in cyber threats demands preventive approaches based on the shift-left security paradigm, positioning Quality Assurance (QA) as a proactive defense at the code level. In this context, a literature review is presented that analyzed 16 primary studies published from 2021 onwards. The results demonstrate that automated testing intercepts logical flaws at their root, disrupting the causal chain of vulnerabilities and reducing remediation costs. It is concluded that fully leveraging this preventive potential mitigates the risk exposure window, yet still depends on overcoming technical and cultural integration barriers among development, QA, and security teams.*

Resumo. *O aumento das ameaças cibernéticas exige abordagens preventivas baseadas no paradigma shift-left security, posicionando a Garantia da Qualidade (Quality Assurance - QA) como defesa proativa no nível do código. Neste contexto, apresenta-se uma revisão de literatura que analisou 16 estudos primários publicados a partir de 2021. Os resultados demonstram que testes automatizados interceptam defeitos lógicos em sua raiz, quebrando a cadeia causal das vulnerabilidades e reduzindo custos de correção. Conclui-se que o pleno aproveitamento desse potencial preventivo mitiga a janela de exposição a riscos, mas ainda depende da superação de barreiras de integração técnica e cultural entre as equipes de desenvolvimento, QA e segurança.*

1. Introdução

O crescimento da dependência de sistemas digitais transformou o software em um ativo crítico para organizações públicas, privadas e acadêmicas. À medida que as aplicações se tornam mais complexas, falhas de implementação, validação e tratamento de erros passam a produzir impactos que ultrapassam a dimensão puramente funcional, alcançando diretamente a segurança dos ecossistemas tecnológicos [Da Conceição 2024].

Tradicionalmente, a segurança foi tratada como etapa posterior ao desenvolvimento, voltada à remediação de vulnerabilidades já identificadas ou exploradas. A engenharia de software contemporânea desloca esse foco para mecanismos preventivos desde as fases iniciais do ciclo de vida, apoiando-se em automação de testes, verificação contínua e integração entre desenvolvimento, operações e segurança, como propõem DevSecOps e *shift-left security* [Kathiresan 2022, NIST 2022].

Neste contexto, o artigo parte da premissa de que a segurança cibernética é atributo intrínseco da qualidade de software, e não camada periférica aplicada ao final do desenvolvimento [Bojanova and Galhardo 2023]. Sob essa ótica, a Garantia da Qualidade (QA)

sustenta a estratégia *shift-left* ao validar requisitos de segurança junto aos funcionais. O argumento central é que o ciberataque consumado representa o estágio final de uma cadeia causal frequentemente iniciada por defeitos de lógica e codificação.

Diante disso, o objetivo deste trabalho é realizar uma revisão de literatura [Kitchenham and Charters 2007] sobre a intersecção entre Garantia da Qualidade e Segurança da Informação. Busca-se compreender como *bugs* e vulnerabilidades se conectam teoricamente, quais práticas de QA atuam de forma preventiva contra ameaças e quais impactos socioorganizacionais e técnicos são relatados quando a segurança é integrada ao ciclo de vida desde a concepção do produto.

2. Referencial Teórico

2.1. Quality Assurance e a Anatomia das Falhas de Software

A QA compreende um conjunto sistemático de práticas voltadas à prevenção de defeitos, verificação de requisitos, validação do comportamento esperado e melhoria contínua dos processos de desenvolvimento. Em ecossistemas modernos de engenharia de software, o escopo de QA transcende a mera execução de testes manuais, englobando revisões técnicas formais, automação de testes, pipelines de integração contínua, observabilidade e análise retrospectiva de falhas [Marques and Silva 2024].

Sob essa perspectiva, as anomalias de software deixam de ser indicadores de baixa qualidade funcional, pois omissões lógicas, validações insuficientes e exceções mal tratadas fundamentam explorações maliciosas. Torna-se imperativo, portanto, distinguir a tríade clássica da engenharia: o **erro (*error*)**, ação humana equivocada na especificação ou codificação; o **defeito (*fault/defect*)**, manifestação estática do erro no artefato; e a **falha (*failure*)**, desvio dinâmico do comportamento esperado do sistema em execução.

A convergência entre qualidade e segurança também se consolida por meio do monitoramento de métricas de software, tais como densidade de defeitos, cobertura de código por testes e o tempo médio de reparo (*Mean Time to Repair* - MTTR). Testes focados em segurança, nesse contexto, expandem as capacidades tradicionais de *software quality assurance*, dado que buscam atestar se as barreiras de confiabilidade do sistema resistem mesmo sob estímulos intencionais e adversariais de violação [Abdi et al. 2012].

2.2. O Modelo Causal de Vulnerabilidades: A Conexão entre QA e Segurança

Este estudo parte da premissa de que vulnerabilidades de segurança decorrem de uma cadeia causal, e não de eventos espontâneos. Um defeito estático no código-fonte pode propagar estados internos inconsistentes durante a execução até se manifestar como uma falha de segurança explorável. Essa relação é formalizada pela taxonomia clássica de vulnerabilidades, que descreve a progressão de um *bug* inicial por sucessivos estados de erro até uma falha de segurança sistêmica (*security failure*) [Bojanova and Galhardo 2023].

A Figura 1 apresenta a relação entre engenharia de software e segurança da informação, mostrando que um erro humano de especificação ou codificação gera um defeito no código, que durante a execução pode causar estados inconsistentes e resultar em uma falha de segurança. Enquanto a segurança tradicional atua de forma reativa sobre falhas já expostas, a QA busca prevenir problemas desde sua origem, interrompendo a cadeia causal antes que o defeito se torne uma vulnerabilidade explorável, em consonância com a abordagem de *shift-left security*.

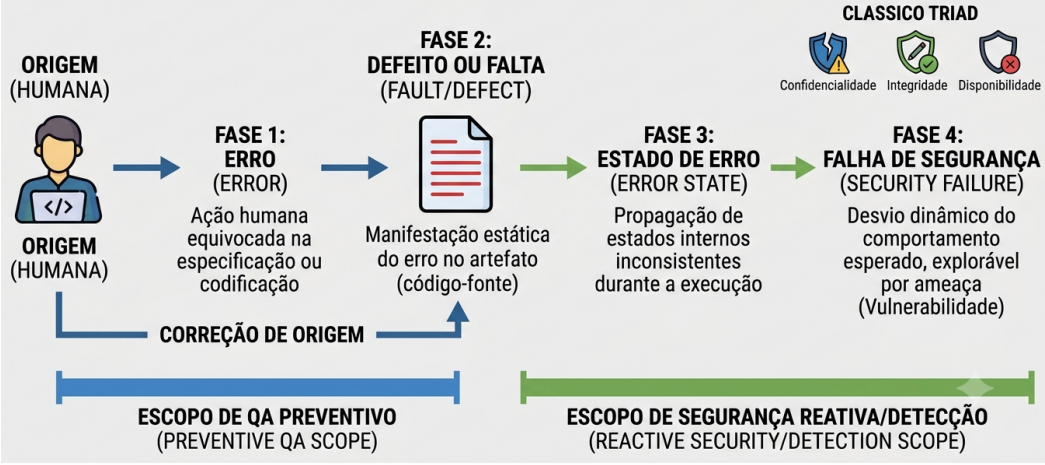


Figura 1. Síntese da cadeia causal entre erro humano, defeito de software, estado de erro e falha de segurança explorável, com a delimitação dos escopos preventivo e reativo.

Essa formulação dá sustentação matemática e lógica à tese defendida neste artigo: se a falha de segurança constitui o estágio final e visível de uma cadeia de dependências técnicas, a interceptação e correção do defeito em sua origem atua de forma cirúrgica na quebra da causalidade. Em contrapartida, mecanismos puramente reativos aplicados ao final do ciclo concentram-se na mitigação da manifestação externa da vulnerabilidade, gerando correções superficiais e de maior custo adaptativo.

Na engenharia de software, a interface entre QA e segurança traduz-se em testes automatizados, sanitização de entradas e validações em esteiras CI/CD. Como vulnerabilidades críticas (*SQL Injection*, XSS e quebras de acesso) decorrem de falhas de validação e desenho, os testes funcionais atuam de forma complementar às ferramentas tradicionais, posicionando o QA como a primeira linha de defesa ativa no nível do código [Nakov et al. 2021].

2.3. Vulnerabilidades de Segurança e a Filosofia *Shift-Left*

No domínio da segurança da informação, vulnerabilidades são fraquezas exploráveis que podem comprometer a confidencialidade, a integridade ou a disponibilidade de um sistema. A abordagem *shift-left security* antecipa sua identificação para as fases iniciais do ciclo de vida de desenvolvimento de software (SDLC), reduzindo custos de correção, retrabalho arquitetural e tempo de exposição ao risco (*window of vulnerability*).

Essa transição está perfeitamente alinhada com as diretrizes do *Secure Software Development Framework* (SSDF) publicado pelo NIST, o qual estrutura o desenvolvimento seguro em quatro pilares fundamentais: preparação da organização, proteção do software, produção de software seguro e resposta a vulnerabilidades residuais. Sob a ótica de QA, o deslocamento preconizado pelo SSDF altera o foco da engenharia de testes: o objetivo deixa de ser apenas ”encontrar vulnerabilidades prontas”e passa a ser o desenho de processos que impeçam a inserção do defeito lógico primário [NIST 2022].

Nas metodologias ágeis contemporâneas, essa lógica é operacionalizada através da cultura DevSecOps. Em substituição ao modelo tradicional que posicionava a segurança da informação como um processo isolado de auditoria tardia pré-homologação, o para-

digma *shift-left* advoga por sua diluição contínua desde a especificação de requisitos até o monitoramento em produção. Consolida-se, assim, a mentalidade de *security-as-quality*, estabelecendo a segurança não como um adendo opcional, mas como um requisito não funcional de qualidade e um atributo indissociável do produto [Kathiresan 2022].

3. Materiais e Métodos

Esta pesquisa configura-se como uma revisão de literatura de caráter descritivo e analítico. O percurso metodológico seguiu o protocolo de planejamento, condução e documentação formalizado por Kitchenham [Kitchenham and Charters 2007], tornando auditáveis e rastreáveis os processos de busca, seleção, avaliação de qualidade, extração de dados e síntese dos estudos que compõem o corpus.

3.1. Planejamento da Revisão

Foram investigadas evidências empíricas e conceituais sobre a interdependência entre testes de software e vulnerabilidades, considerando os processos de qualidade como defesa preventiva no paradigma *shift-left*. Três Questões de Pesquisa (RQs) orientaram a condução da revisão e os eixos temáticos apresentados nos Resultados.

RQ1: Como a literatura científica correlaciona ou distingue os conceitos de falhas de software (*bugs*) e vulnerabilidades de segurança?
Eixo de análise: Cadeia causal entre defeito lógico e falha de segurança.

RQ2: Quais práticas específicas de *Quality Assurance* são evidenciadas como mecanismos proativos e preventivos de segurança?
Eixo de análise: Práticas de teste, automação e esteiras DevSecOps/shift-left.

RQ3: Quais impactos e benefícios técnicos ou organizacionais são reportados ao integrar as práticas de QA à abordagem de *shift-left security*?
Eixo de análise: Métricas integradas de segurança e qualidade.

A estratégia de busca automatizada foi executada em quatro bases relevantes para Engenharia de Software e Segurança da Informação: IEEE Digital Library, Biblioteca Digital da Sociedade Brasileira de Computação, Biblioteca Digital Brasileira de Teses e Dissertações (BDTD) e Google Acadêmico.

A *string* de busca principal foi construída a partir do cruzamento de termos booleanos (AND/OR) em inglês e português, cobrindo os eixos de engenharia de processos, testes e cibersegurança: *(“Software engineering”OR “Engenharia de software”OR “Software development”OR “Desenvolvimento de software”OR “CI/CD”) AND (“Quality Assurance”OR “Garantia da qualidade”OR “Software testing”OR “Testes de software”OR “Prevenção de falhas”) AND (“Cybersecurity”OR “Cibersegurança”OR “Software vulnerabilities”OR “Vulnerabilidades de software”)*

A definição do corpus primário seguiu critérios explícitos de elegibilidade. Foram incluídos artigos científicos e revisões publicados desde 2021, em inglês ou português, que abordassem diretamente a mitigação de vulnerabilidades ou a prevenção de erros por meio de QA, testes ou *shift-left*. Definido previamente no protocolo, o marco de 2021 equilibra atualidade e abrangência: privilegia evidências produzidas durante a consolidação recente de DevSecOps e permite observar estudos imediatamente anteriores e posteriores ao SSDF 1.1, de 2022, sem combinar esse cenário com contextos tecnológicos substancialmente mais antigos [NIST 2022].

Foram excluídos estudos desalinhados ao escopo, duplicados, sem acesso ao texto completo ou restritos a *machine learning* e computação em nuvem sem nexo explícito

com a qualidade do ciclo de desenvolvimento. O CE 4 evita contabilizar como evidência sobre QA pesquisas centradas apenas no desempenho de modelos ou na configuração de infraestrutura. Portanto, o critério não exclui essas tecnologias de maneira geral: aplicações voltadas a testes, prevenção de vulnerabilidades ou melhoria do SDLC permaneceram elegíveis por contribuírem diretamente para as RQs. Estudos seminais anteriores a 2021 foram preservados somente para fundamentação conceitual. A Tabela 1 sumariza os parâmetros.

Tabela 1. Critérios de inclusão (CI) e exclusão (CE) dos estudos

Inclusão	**Exclusão**
CI 1 - Publicações compreendidas a partir do ano de 2021;	CE 1 - Estudos tematicamente desalinhados à convergência entre QA e segurança;
CI 2 - Textos redigidos integralmente nos idiomas inglês ou português;	CE 2 - Artigos identificados como duplicados entre as bases de dados;
CI 3 - Abordagem direta sobre prevenção de erros ou mitigação de vulnerabilidades no ciclo de vida;	CE 3 - Trabalhos focados estritamente em infraestrutura de cibersegurança sem correlação com desenvolvimento;
CI 4 - Discussão explícita de *Quality Assurance*, técnicas de teste ou filosofia *shift-left*;	CE 4 - Pesquisas restritas a *machine learning* ou computação em nuvem isoladas de processos de testes.
CI 5 - Disponibilidade de acesso ao texto completo nas plataformas consultadas.	

3.2. Condução e Seleção dos Estudos

O processo de condução operacionalizou-se por um fluxo de afunilamento progressivo em sete etapas, cujas métricas quantitativas por base são consolidadas na Tabela 2:

1. **Busca Inicial:** recuperação dos registros retornados pela aplicação da *string* nas bases selecionadas.
2. **Descarte 1:** triagem de títulos, resumos e palavras-chave para remover duplicatas (**CE 2**) e desvios temáticos evidentes (**CE 1**, **CE 3** e **CE 4**).
3. **Seleção Preliminar:** agrupamento dos estudos remanescentes considerados potencialmente elegíveis.
4. **Descarte 2:** leitura da introdução, método e conclusões para verificar período e idioma (**CI 1–2**), aderência ao escopo (**CI 3–4**) e fatores impeditivos (**CE 1**, **CE 3–4**).
5. **Seleção Secundária:** encaminhamento dos estudos aprovados à avaliação de qualidade metodológica.
6. **Descarte 3:** exclusão de produções com qualidade insuficiente ou sem acesso ao texto completo (**CI 5**).
7. **Seleção Final:** formação do corpus destinado à extração semântica e à síntese das evidências.

Os artefatos finais foram catalogados em matriz de extração com autoria, ano, natureza da evidência, práticas de QA, riscos mitigados e principais achados.

3.3. Critérios de Qualidade (CQ) Metodológica

Os estudos da seleção secundária foram avaliados em cinco dimensões: CQ 1) presença de evidências empíricas ou métricas quantitativas; CQ 2) proposição de soluções aplicáveis à rotina operacional de QA; CQ 3) delimitação clara dos objetivos associados a testes e segurança; CQ 4) descrição do cenário ou contexto organizacional; e CQ 5) alinhamento metodológico com estratégias de prevenção de anomalias lógicas.

Adotou-se uma escala triádica (1,0 pleno; 0,5 parcial; 0,0 nulo), atribuindo-se 0,5 a estudos conceituais que, mesmo sem validação empírica ou estatística completa, agregavam valor à pesquisa. O somatório produziu um índice percentual de qualidade, de 0 a 5,0, que orientou o Descarte 3.

O protocolo e a triagem do *corpus* foram gerenciados na plataforma *Parsifal*. Os modelos generativos *ChatGPT* e *Gemini* foram empregados exclusivamente como suporte à revisão gramatical, à formatação em LaTeX e à concepção de ilustrações. A análise crítica, a extração semântica e a síntese das evidências permaneceram sob responsabilidade intelectual das pessoas autoras.

4. Resultados e Discussão

Após a execução das estratégias de busca, aplicação dos critérios de elegibilidade e extração sistemática dos dados, os resultados foram catalogados e sintetizados para responder às questões de pesquisa estabelecidas. A Tabela 2 sintetiza o fluxo de afunilamento quantitativo dos estudos recuperados nas bases consultadas.

Tabela 2. Síntese do processo de seleção e afunilamento dos estudos

Base de Dados	Busca	Descarte 1	Sel. Prelim.	Descarte 2	Sel. Secund.	Sel. Final
IEEE Digital Library	113	84	29	14	15	6
Google Acadêmico	200	121	79	44	35	10
BDTD	19	19	0	–	–	0
Periódicos SBC	3	3	0	–	–	0
Total Geral	**335**	**227**	**108**	**58**	**50**	**16**

A avaliação dos 16 estudos utilizou os cinco critérios metodológicos definidos, pontuados em 1,0 (pleno), 0,5 (parcial) ou 0,0 (nulo). A Tabela 3 reúne os escores e a correlação com as RQs; o valor 0,5 identifica estudos conceituais ou arquiteturais relevantes, ainda que sem validação empírica completa.

Tabela 3. Avaliação da qualidade metodológica e correlação do corpus com as RQs

Estudo Selecionado	CQ 1	CQ 2	CQ 3	CQ 4	CQ 5	%	RQ(s)
[Bojanova and Galhardo 2023]	0,5	0,5	1,0	1,0	0,5	70%	RQ1
[Nakov et al. 2021]	0,5	1,0	1,0	1,0	1,0	90%	RQ1, RQ2
[Corradini et al. 2023]	1,0	1,0	1,0	1,0	1,0	100%	RQ1, RQ2
[Cunha and Pombo 2023]	1,0	1,0	1,0	1,0	1,0	100%	RQ1, RQ2
[Andrade et al. 2023]	1,0	1,0	1,0	1,0	1,0	100%	RQ1
[Elias 2023]	0,5	1,0	0,5	0,5	0,5	60%	RQ2
[Farias et al. 2025]	1,0	0,5	0,5	1,0	1,0	80%	RQ2, RQ3
[Kaniewski et al. 2025]	1,0	0,5	1,0	1,0	1,0	90%	RQ1
[Kathiresan 2022]	1,0	1,0	1,0	1,0	1,0	100%	RQ2, RQ3
[Kathiresan 2024]	0,5	1,0	1,0	0,5	0,5	70%	RQ2
[Kathiresan 2025]	1,0	1,0	1,0	1,0	1,0	100%	RQ3
[Ojuri 2021]	0,5	1,0	1,0	0,5	0,5	70%	RQ3
[Ojuri 2023]	0,5	1,0	1,0	0,5	0,5	70%	RQ1, RQ2
[Ojuri 2024]	0,5	1,0	1,0	0,5	0,5	70%	RQ3
[Ramdass 2024]	0,5	1,0	1,0	0,5	1,0	80%	RQ2
[Nourry et al. 2024]	1,0	1,0	1,0	1,0	1,0	100%	RQ1, RQ2
Média Geral	**0,75**	**0,91**	**0,94**	**0,81**	**0,78**	**83,75%**	–

Os indicadores demonstram qualidade metodológica satisfatória, com média de 83,75%. CQ 3 (clareza do escopo) e CQ 2 (aplicabilidade) obtiveram maior aderência; as principais limitações foram a ausência de métricas empíricas em parte do corpus (CQ 1) e o menor detalhamento de estudos conceituais. Ainda assim, sua convergência forneceu base suficiente para a síntese e a coluna RQ(s) explicita a rastreabilidade das evidências.

4.1. A Relação Epistêmica entre Falhas de Software e Vulnerabilidades (RQ1)

As evidências demonstram que falhas de software e vulnerabilidades constituem construtos teóricos distintos, embora integrados ao mesmo contínuo técnico. Sob o prisma

da literatura, o *bug* é um defeito de especificação ou codificação; o erro reflete o desvio comportamental em tempo de execução; e a vulnerabilidade é a exposição explorável dessa inconsistência por uma ameaça. Essa dissociação corrobora as distinções clássicas entre *fault, bug, error, weakness* e *vulnerability*, evidenciando que a quebra de segurança resulta de anomalias cumulativas de código, falhas de design e de integração semântica [Bojanova and Galhardo 2023, Nakov et al. 2021].

O arcabouço normativo estabelecido pelo SSDF do NIST dá suporte a essa interpretação ao compartimentar o ciclo de vida seguro em fases que mitigam proativamente a superfície de ataque. Abordagens de segurança estritamente tradicionais e reativas operam no limite extremo da cadeia causal, agindo apenas após a identificação pública, exploração ativa ou requisição emergencial de correções (*patches*).

A tese central defendida neste estudo propõe que a engenharia de QA reposicione essas intervenções para as fases embrionárias do ciclo, interceptando defeitos lógicos antes que estes se convertam em vulnerabilidades latentes no ambiente de produção [NIST 2022].

Essa dinâmica ganha contornos práticos nas evidências coletadas. Vulnerabilidades críticas de atribuição em massa (*mass assignment*) em APIs REST decorrem de falhas de configuração e amarração de tipos em *frameworks*; quebras estruturais em verificações de permissões originam-se da má interpretação de padrões arquiteturais seguros; e comportamentos anômalos expostos por testes de mutação ou *fuzzing* derivam de dados de entrada imprevistos na matriz de testes determinísticos.

Por conseguinte, em resposta à RQ1, conclui-se que falhas e vulnerabilidades estão ligadas por um nexo de causalidade técnica: a vulnerabilidade de cibersegurança é a manifestação explorável de uma falha latente de qualidade de software que não foi mitigada em tempo oportuno [Corradini et al. 2023, Cunha and Pombo 2023, Nourry et al. 2024, Andrade et al. 2023, Kaniewski et al. 2025, Ojuri 2023].

4.2. Práticas de Quality Assurance como Mecanismos Proativos de Defesa (RQ2)

As práticas de QA como vetores de segurança ativa envolvem testes automatizados, higienização de entradas, revisões de código, análises estáticas (SAST) e dinâmicas (DAST), testes de APIs e validação em esteiras CI/CD. A execução conjunta dessas técnicas materializa as diretrizes do SSDF, distribuindo os controles entre as fases de preparação e codificação protegida, reduzindo a dependência histórica de auditorias tardias pré-homologação [NIST 2022, Nakov et al. 2021, Kathiresan 2022].

A literatura selecionada confere especial relevância a três abordagens emergentes: o teste automatizado focado nas vulnerabilidades lógicas de APIs, o desenvolvimento de suítes de testes orientadas a padrões arquiteturais de segurança (*security patterns*) e a aplicação de testes exploratórios automatizados via *fuzzing*. Complementarmente, constata-se a inserção progressiva de técnicas avançadas de inteligência artificial generativa em QA para automação de testes de regressão, priorização de criticidade de vulnerabilidades e geração preditiva de cenários de teste adversariais em pipelines DevSecOps.

A RQ2 é respondida, portanto, pela consolidação de uma matriz de testes multifacetada e integrada, que transforma o analista de QA em um agente ativo na execução técnica do paradigma *shift-left* [Corradini et al. 2023, Cunha and Pombo 2023, Ramdass 2024, Elias 2023, Kathiresan 2024, Ojuri 2023, Nourry et al. 2024].

Ademais, os achados destacam que a eficácia dessas ferramentas está associada às competências analíticas humanas do time de QA. Atributos cognitivos como pensamento crítico adversarial, comunicação transdisciplinar e capacidade de depuração complexa permitem que os engenheiros de testes interpretem ”sinais fracos”na arquitetura antes que estes evoluam para fraquezas sistêmicas exploráveis, reforçando o QA como a primeira barreira de defesa do código [Farias et al. 2025].

4.3. Impactos Técnicos e Organizacionais do Shift-Left Baseado em QA (RQ3)

Os impactos decorrentes da convergência entre QA e *shift-left security* englobam a antecipação cronológica da descoberta de falhas, a contração expressiva da janela de exposição a riscos, a redução de retrabalho arquitetural e a otimização do tempo médio de remediação (MTTR). Essa transição metodológica altera a postura da governança de TI, migrando de um modelo reativo de contenção de incidentes para uma arquitetura preventiva de resiliência de software [NIST 2022, Abdi et al. 2012, Kathiresan 2022].

O monitoramento de métricas integradas surge como fator crítico para atestar esses benefícios. Indicadores como a densidade de defeitos de segurança, a cobertura de código por testes adversariais, a taxa de regressão de vulnerabilidades e o tempo de ciclo para correção em desenvolvimento demonstram empiricamente a eficácia da abordagem. Sob essa ótica, o desenvolvimento de remendos de segurança (*patches*) em produção passa a ser interpretado como um indicador de falha de escape do QA, evidenciando que os mecanismos preventivos automáticos falharam nas esteiras iniciais do desenvolvimento [Ojuri 2021, Abdi et al. 2012].

Por fim, a hibridização de padrões internacionais de conformidade com os modelos operacionais de QA potencializa esse impacto. Ao fundir os critérios de qualidade da norma ISO/IEC 25010 com os requisitos de segurança da ISO/IEC 27001, do NIST Cybersecurity Framework e dos guias OWASP, a engenharia contemporânea estabelece a segurança como um Critério de Aceite (*Definition of Done*) nativo.

Assim, em resposta à RQ3, conclui-se que o QA integrado ao *shift-left* atenua sensivelmente os custos e riscos associados a vulnerabilidades, contanto que amparado por automação contínua, taxonomia de riscos clara e alinhamento cultural entre desenvolvimento, QA e segurança [Ojuri 2024, NIST 2022, Farias et al. 2025, Kathiresan 2025].

4.4. Ameaças à Validade do Estudo

As principais limitações e ameaças à validade desta revisão concentram-se em quatro dimensões: validade de constructo, seleção de estudos, avaliação qualitativa e validade externa. No âmbito do constructo, a variação terminológica entre os conceitos de *bug, bug de segurança* e *vulnerabilidade* na literatura foi mitigada pela adoção estrita do mapeamento causal padronizado nesta pesquisa, evitando falsas equivalências conceituais. Quanto à seleção, restrições inerentes aos indexadores, idiomas e escopo da *string* booleana podem ter ocultado artigos relevantes; risco mitigado pela transparência e auditabilidade do protocolo documentado no Parsifal.

A avaliação de qualidade foi conduzida pelos autores, sem arbitragem independente, restringindo os percentuais da Tabela 3 a indicadores internos. Além disso, a heterogeneidade dos domínios analisados e o recorte temporal limitam a generalização quantitativa dos achados para cenários corporativos genéricos.

5. Conclusão

Este trabalho investigou a intersecção entre práticas de *Quality Assurance* e Segurança da Informação, evidenciando que falhas de software e vulnerabilidades formam um contínuo técnico-causal. Os achados indicam que anomalias de especificação e lacunas de validação funcional são precursores diretos de superfícies exploráveis. Desse modo, conclui-se que o escopo de QA transcende a conformidade, atuando como um instrumento estratégico e preventivo de cibersegurança.

No que tange à generalização dos eixos analíticos, a convergência das evidências permitiu mapear conclusões sólidas para as três questões de pesquisa delineadas:

- **RQ1:** Confirmou-se o nexo causal no qual falhas de lógica de software funcionam como a causa raiz estática de vulnerabilidades dinâmicas, validando o modelo de encadeamento conceitual;
- **RQ2:** Catalogou-se uma matriz robusta de práticas proativas, com destaque para a automação de testes de APIs, testes de regressão de segurança, técnicas de *fuzzing* e a orquestração de ferramentas SAST/DAST em esteiras CI/CD;
- **RQ3:** Mensuraram-se impactos técnicos e organizacionais expressivos, expressos pela contração da janela de exposição a riscos, mitigação do retrabalho arquitetural e otimização do tempo médio de remediação (MTTR).

Frente ao exposto, conclui-se que a Garantia da Qualidade, quando integrada ao *shift-left security*, consolida-se como a primeira linha de defesa do ecossistema tecnológico. Operacionalizar essa filosofia exige tratar a segurança como um requisito não funcional testável (com *Definition of Done* claro) e uma responsabilidade técnica compartilhada entre desenvolvimento, QA e segurança. Para trabalhos futuros, recomenda-se estudos empíricos longitudinais que quantifiquem o impacto da maturidade de QA na densidade de vulnerabilidades residuais em larga escala.

Referências

Abdi, A., Souzani, A., Amirfakhri, M., and Moghadam, A. B. (2012). Using security metrics in software quality assurance process. In *6th International Symposium on Telecommunications (IST'2012)*. IEEE.

Andrade, R., Torres, J., Ortiz-Garcés, I., Miño, J., and Almeida, L. (2023). An exploratory study gathering security requirements for the software development process. *Electronics*, 12(17):3594.

Bojanova, I. and Galhardo, C. E. C. (2023). Bug, fault, error, or weakness: Demystifying software security vulnerabilities. *IT Professional*. Preprint.

Corradini, D., Pasqua, M., and Ceccato, M. (2023). Automated black-box testing of mass assignment vulnerabilities in restful apis. In *45th IEEE/ACM International Conference on Software Engineering (ICSE 2023)*. IEEE/ACM.

Cunha, C. and Pombo, N. (2023). Automated reusable tests for mitigating secure pattern interpretation errors. *IEEE Access*.

Da Conceição, I. T. (2024). Práticas de segurança no desenvolvimento com o uso de metodologias ágeis. *Revista Ibero-Americana de Humanidades, Ciências e Educação*, 10(3):734–754.

Elias, D. (2023). Static analysis in embedded and electronic systems: Challenges, applications, and future perspectives. *International Journal of Modern Computing*.

Farias, R. S., Ahmed, I., and Almeida, E. S. (2025). What makes a great software quality assurance engineer? *IEEE Transactions on Software Engineering*, 51(4):1153–.

Kaniewski, S., Schmidt, F., Enzweiler, M., Menth, M., and Heer, T. (2025). A systematic literature review on detecting software vulnerabilities with large language models. *ACM Transactions on Software Engineering and Methodology*.

Kathiresan, G. (2022). Evaluating the impact of devsecops on software quality: A systematic review and empirical study. *World Journal of Advanced Research and Reviews*, 14(1):644–653.

Kathiresan, G. (2024). A multimodal approach to software quality assurance: Integrating static analysis, dynamic testing, and ai-based anomaly detection. *International Journal of Innovative Research in Computer and Communication Engineering*, 12(2).

Kathiresan, G. (2025). Resilience by design: Embedding cyber-attack scenarios into qa workflows for zero-day readiness. *International Journal of Communication Networks and Information Security*, 17(2).

Kitchenham, B. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE 2007-001, Keele University and Durham University Joint Report, Keele, UK.

Marques, L. F. d. F. and Silva, R. d. O. F. d. (2024). Quality assurance: a eficiência dos testes para elevar a qualidade e o desempenho do sistema. Trabalho de Conclusão de Curso, Curso Superior de Tecnologia em Tecnologia da Informação, Faculdade de Tecnologia de Assis. Orientador: Prof. Dr. José Luiz Guimarães.

Nakov, O., Trifonov, R., Nakov, P., and Pavlova, G. (2021). Analysis of software vulnerabilities, measures for prevention and protection and security testing. In *29th National Conference with International Participation "Telecom 2021"*. IEEE.

NIST (2022). National Institute of Standards and Technology. Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities. Special Publication NIST SP 800-218, National Institute of Standards and Technology.

Nourry, O., Kondo, M., Alfadel, M., McIntosh, S., and Kamei, Y. (2024). Exploring the adoption of fuzz testing in open-source software: A case study of the go community. In *2024 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 300–311. IEEE.

Ojuri, M. A. (2021). Evaluating cybersecurity patch management through qa performance indicators. *International Journal of Technology Management & Humanities*, 7(4):30–.

Ojuri, M. A. (2023). Ai-driven quality assurance for secure software development lifecycles. *International Journal of Technology Management & Humanities*, 9(1):25–.

Ojuri, M. A. (2024). Integrating cybersecurity standards into software quality assurance frameworks: A holistic approach. *Journal of Economics, Finance and Accounting Studies*, pages 258–.

Ramdass, K. (2024). Enhancing dynamic application security testing (dast): A systematic literature review. Preprint, Research Square.