Universidade de Brasília – UnB
Faculdade de Ciências e Tecnologia em Engenharia – FCTE
Engenharia de Software

Boas Práticas de Segurança de Software Orientado para *Startups* a Partir de uma Revisão Multi-vocal de Literatura

Autor: Carlos Eduardo de Sousa Fiuza
Orientadora: Profª. Drª. Elaine Venson

Brasília, DF
2025

Carlos Eduardo de Sousa Fiuza

Boas Práticas de Segurança de Software Orientado para *Startups* a Partir de uma Revisão Multi-vocal de Literatura

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB

Faculdade de Ciências e Tecnologia em Engenharia – FCTE

Orientador: Profª. Drª. Elaine Venson

Brasília, DF

2025

Carlos Eduardo de Sousa Fiuza

Boas Práticas de Segurança de Software Orientado para *Startups* a Partir de uma Revisão Multi-vocal de Literatura/ Carlos Eduardo de Sousa Fiuza. – Brasília, DF, 2025- 116 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof[a]. Dr[a]. Elaine Venson

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade de Ciências e Tecnologia em Engenharia – FCTE , 2025.

1. Práticas de segurança de software. 2. Startup. I. Prof[a]. Dr[a]. Elaine Venson. II. III. Universidade de Brasília. IV. Faculdade de Ciências e Tecnologia em Engenharia - FCTE. V. Boas Práticas de Segurança de Software Orientado para *Startups* a Partir de uma Revisão Multi-vocal de Literatura

CDU 02:141:005.6

Carlos Eduardo de Sousa Fiuza

Boas Práticas de Segurança de Software Orientado para *Startups* a Partir de uma Revisão Multi-vocal de Literatura

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 17 de Fevereiro de 2025:

Profª. Drª. Elaine Venson
Orientador

Prof. Dr. Renato Coral Sampaio
Convidado 1

Lucas Gabriel Bezerra
Convidado 2

Brasília, DF
2025

A todos que acreditaram em mim, mesmo nos momentos em que eu duvidava.
Sua confiança foi um impulso fundamental para minha perseverança.

Agradecimentos

Gostaria de expressar meus sinceros agradecimentos a algumas pessoas especiais que foram fundamentais ao longo desta jornada acadêmica. Em primeiro lugar, minha querida mãe, Ivane, cujo amor incondicional e apoio constante foram minha rocha durante os desafios deste percurso. Sua dedicação à minha educação é a luz que me guiou, e este trabalho é dedicado a você, que sempre acreditou em meu potencial. A minha amada irmã, Camila, também merece meu profundo agradecimento. Sua presença e encorajamento foram fontes constantes de inspiração, e compartilhar este momento contigo é motivo de grande alegria.

Não poderia deixar de expressar minha gratidão à minha incrível namorada, Nathália, cujo apoio inabalável e compreensão tornaram os dias mais desafiadores mais leves e as vitórias mais significativas. Seu amor é a motivação por trás de cada conquista, e agradeço por caminharmos juntos nesta jornada. À minha dedicada orientadora, Elaine, que guiou meus passos com paciência, sabedoria e um compromisso incansável com a excelência acadêmica. Suas orientações foram essenciais para o desenvolvimento deste trabalho, e sou grato pela oportunidade de aprender com uma profissional tão dedicada.

Por fim, expresso minha gratidão à Universidade de Brasília, que não apenas me proporcionou uma educação de qualidade, mas também um ambiente enriquecedor repleto de professores excepcionais. Agradeço por me permitir cursar um programa tão completo, que ampliou minha visão de mundo e me preparou para os desafios futuros. Este trabalho é um reflexo do aprendizado valioso que recebi nesta instituição notável.

"Se fiz descobertas valiosas, foi mais por ter paciência do que qualquer outro talento."
(Isaac Newton)

Resumo

A adoção de práticas de segurança no desenvolvimento de software se tornou imprescindível nos últimos anos; clientes, usuários e entidades regulamentadoras requerem softwares cada vez mais seguros. As organizações, em específico as *startups* de software, tem buscado a adoção de práticas para o desenvolvimento de produtos seguros, mas encontram diversas dificuldades, como escassa literatura acerca do tema e recursos extremamente limitados para aplicação de estratégias que demandam treinamento e profissionais qualificados. O objetivo deste trabalho é a identificação de boas práticas de segurança para *startups* de software, em especial àquelas em estágio inicial, com práticas categorizadas e vinculadas para seu específico contexto. A metodologia é do tipo descritiva com emprego da técnica de revisão multi-vocal de literatura qualitativa, aplicada em cinco etapas. A primeira compreendeu o processo de busca automatizado e manual, seguido da seleção de estudos por critérios de inclusão e exclusão, avaliação de qualidade por *checklist*, extração dos dados por formulários e por fim uma síntese que utilizou o processo de análise temática qualitativa. As práticas de segurança de software que apareceram com maior frequência são às pertencentes a área de conhecimento Ciclo de Vida Seguro de Software, seguida da área de Aspectos Humanos, Organizacionais e Regulatórios. Os desafios de implementação apurados das práticas se concentraram em profissionais qualificados, dificuldade de implementação, conhecimento técnico e divulgação de conhecimento. O estudo apresentou boas práticas de segurança aplicáveis a *startups* categorizadas por área de conhecimento.

Palavras-chave: Segurança de Software; Startup; Engenharia de Software; Revisão Multivocal de Literatura; Práticas de Segurança.

Abstract

The adoption of security practices in software development has become essential in recent years, as clients, users, and regulatory entities increasingly demand more secure software. Organizations, particularly software startups, have been striving to implement practices for the development of secure products, but they face various challenges, such as limited literature on the subject and extremely constrained resources for the implementation of strategies that require training and qualified professionals. The objective of this study is to identify best security practices tailored for software startups, especially those in the early stages, with practices categorized and linked to their specific context. The methodology employed is descriptive, utilizing the qualitative multivocal literature review technique, applied in five stages. The first comprised the automated and manual search process, followed by the selection of studies by inclusion and exclusion criteria, quality assessment by checklist, data extraction by forms and finally a synthesis that used the qualitative thematic analysis process. The software security practices that appeared most frequently are those belonging to the Secure Software Life Cycle knowledge area, followed by the Human, Organizational and Regulatory Aspects area. The implementation challenges identified for the practices focused on qualified professionals, implementation difficulties, technical knowledge and knowledge dissemination. The study presented good security practices applicable to startups categorized by knowledge area.

Keywords: Software Security; Startup; Software Engineering; Multivocal Literature Review; Security Practices.

Lista de ilustrações

Figura 1 – Áreas de conhecimento do escopo CyBOK. Fonte: CyBOK 33
Figura 2 – Gerenciamento de Controle de Segurança de Aplicativos. Fonte: SAFECode (SAFECODE, 2018) 41
Figura 3 – "Tons" de LC. Fonte: Garousi, Felderer e Mäntylä (2019) 52
Figura 4 – Evidências coletadas por base de dados. Fonte: Autor 60
Figura 5 – Evidências por ano. Fonte: Autor 60
Figura 6 – Evidências selecionadas e aceitas por base de dados. Fonte: Autor . . . 61
Figura 7 – Práticas agrupadas por áreas de conhecimento de baixo nível CyBoK. Fonte: Autor 64
Figura 8 – Práticas combinadas e agrupadas por áreas de conhecimento de baixo nível CyBOK. Fonte: Autor 66
Figura 9 – Dificuldades agrupadas por tema. Fonte: Autor 68
Figura 10 – Contexto de aplicação das práticas de segurança. Fonte: Autor 69
Figura 11 – Práticas combinadas e agrupadas por áreas de conhecimento de alto nível CyBOK. Fonte: Autor 70

Lista de quadros

Quadro 1 – Questões para decidir quando adotar LC na revisão adaptado de Garousi, Felderer e Mäntylä (2019) 46
Quadro 2 – Estudos do Conjunto Ouro 48
Quadro 3 – Aplicação do *framework* PICOC 48

Lista de tabelas

Tabela 1 – Primeira versão do conjunto de palavras-chave 49
Tabela 2 – Versão final do conjunto de palavras-chave 49
Tabela 3 – Versão final da *String* de busca 50
Tabela 4 – Critérios de Seleção de Estudo 52
Tabela 5 – Conceitos de Qualidade adaptado de Kitchenham, Budgen e Brereton (2016) 53
Tabela 6 – Definição e relação entre áreas empíricas de pesquisa, por Dybå e Dingsøyr (2008) e conceitos de qualidade por Kitchenham, Budgen e Brereton (2016) 54
Tabela 7 – *Checklist* da avaliação da qualidade 57
Tabela 8 – Formulário de extração de dados 58
Tabela 9 – Resultado da Avaliação de Qualidade 62
Tabela 10 – Relação entre base de dados e quantitativo com score maior ou igual a 5.0 63
Tabela 11 – Aplicação da primeira versão da *string* de busca 80
Tabela 12 – Extração dos Dados E1 81
Tabela 13 – Extração dos Dados E2 83
Tabela 14 – Extração dos Dados E5 85
Tabela 15 – Extração dos Dados E6 86
Tabela 16 – Extração dos Dados E7 88
Tabela 17 – Extração dos Dados E9 89
Tabela 18 – Extração dos Dados E10 90
Tabela 19 – Extração dos Dados E11 93
Tabela 20 – Extração dos Dados E13 95
Tabela 21 – Extração dos Dados E14 97
Tabela 22 – Extração dos Dados E15 98
Tabela 23 – Extração dos Dados E18 100
Tabela 24 – Extração dos Dados E21 102
Tabela 25 – Extração dos Dados E23 106
Tabela 26 – Extração dos Dados E24 107
Tabela 27 – Extração dos Dados E26 109
Tabela 28 – Extração dos Dados E27 111
Tabela 29 – Extração dos Dados E29 113
Tabela 30 – Extração dos Dados E31 115

Lista de abreviaturas e siglas

API	*Application Programming Interface*
ASC	*Application Security Controls*
CD	*Continuos Delivery*
CI	*Continuos Integration*
CSS	*Cascading Style Sheets*
CyBOK	*Cyber Security Body of Knowledge*
E	Evidência
EX	Critério de exclusão
HTML	*HyperText Markup Language*
ID	Identificador
IN	Critério de inclusão
KPI	*Key Performance Indicators*
LC	Literatura Cinza
LF	Literatura Formal
MVP	*Minimum Viable Product*
OE	Objetivo Específico
OG	Objetivo Geral
OWASP	*Open Web Application Security Project*
QP	Questão de Pesquisa
RSL	Revisão Sistemática de Literatura
RML	Revisão Multi-vocal de Literatura
SAFECode	*Software Assurance Forum for Excellence in Code*
SDL	*Microsoft Security Development Lifecycle*
SQL	*Structured Query Language*
XPath	*XML Path Language*

Sumário

1 INTRODUÇÃO . . . 28
1.1 Contexto . . . 28
1.2 Problema . . . 28
1.3 Objetivos . . . 29
1.3.1 Objetivo Geral . . . 29
1.3.2 Objetivos Específicos . . . 29
1.4 Metodologia . . . 29
1.5 Organização do Trabalho . . . 30

2 REFERENCIAL TEÓRICO . . . 31
2.1 Startups de Software . . . 31
2.2 Segurança de Software e Plataforma . . . 33
2.2.1 Segurança de Software . . . 34
2.2.1.1 Categorias de Vulnerabilidades . . . 34
2.2.1.2 Prevenção de Vulnerabilidades . . . 36
2.2.1.3 Detecção de Vulnerabilidades . . . 37
2.2.1.4 Mitigação de Exploração de Vulnerabilidades . . . 37
2.2.2 Ciclo de Vida de Software Seguro . . . 38
2.2.2.1 Microsoft *Security Development Lifecycle* (SDL) . . . 38
2.2.2.2 Software Assurance Forum for Excellence in Code (SAFECode) . . . 41
2.3 Trabalhos Relacionados . . . 42

3 METODOLOGIA . . . 44
3.1 Diretrizes . . . 44
3.2 Planejamento da Revisão . . . 45
3.2.1 Necessidade de uma RML . . . 45
3.2.2 Questões de Pesquisa . . . 46
3.2.3 Estratégia de Busca . . . 46
3.2.3.1 Bases . . . 47
3.2.3.2 Definição de Parada . . . 47
3.2.3.3 Conjunto Ouro . . . 47
3.2.3.4 PICOC . . . 48
3.2.3.5 Palavras-chave e Sinônimos . . . 48
3.2.3.6 *String* de Busca . . . 49
3.2.3.7 Busca Manual por Conferência . . . 50
3.2.4 Seleção de Estudos . . . 51

3.2.5 Avaliação de Qualidade . . . 53
3.2.6 Extração dos Dados . . . 55
3.2.7 Síntese . . . 55

4 RESULTADOS . . . 59
4.1 Processo de Busca . . . 59
4.2 Seleção dos estudos . . . 59
4.3 Avaliação da Qualidade . . . 61
4.4 Extração dos dados . . . 61
4.5 Síntese . . . 62
4.5.1 Organização e Examinação . . . 63
4.5.2 Análise para redução de sobreposições e Definição de temas . . . 63
4.5.3 Criação de Modelo entre temas de ordem superior . . . 67
4.6 Análise . . . 67

5 CONCLUSÃO . . . 72
5.1 Limitações do Trabalho . . . 72
5.2 Trabalhos Futuros . . . 73

REFERÊNCIAS . . . 74

APÊNDICE A – VERSÃO INICIAL *STRING* DE BUSCA . . . 80

APÊNDICE B – DADOS EXTRAÍDOS . . . 81

1 Introdução

1.1 Contexto

Startups de software diferem de pequenas e grandes empresas de várias maneiras, principalmente pela inovação e alta reatividade a mudanças (KEKüLLüOğLU; ACAR, 2023), exigindo investigação, como é demonstrado pelo crescimento no número de pesquisas sobre o desenvolvimento de software nas mesmas (BERG *et al.*, 2018).

Startups surgem a partir de ideias inovadoras (O.J.; SWADIMATH, 2021), sendo uma importante fonte de inovação em software, desenvolvendo produtos únicos orientados aos requisitos do mercado do momento, atuando como vanguarda na aplicação de novas tecnologias (BERG *et al.*, 2018). As que aplicam o modelo *Lean Startup*, empregam um rápido desenvolvimento do *Minimum Viable Product* (MVP), para que suas principais funcionalidades e regras de negócio possam ser disponibilizados para seu público alvo, validando assim sua ideia de negócio e aprimorando futuros incrementos do MVP (SOUZA; CICO; MACHADO, 2021).

A engenharia de software aplicada nas *startups* visa o desenvolvimento bem sucedido de sistemas, usando abordagens sistemáticas que variam o grau de aplicação do processo de acordo a complexidade do sistema, risco de negócio e número de pessoas envolvidas (BERG *et al.*, 2018). Devido ao caráter evolucional de desenvolvimento do produto nas *startups*, a adoção de boas práticas de desenvolvimento de software é prejudicada (SOUZA; CICO; MACHADO, 2021), e sua implantação, de forma inadequada, é um fator significante na taxa de falhas (KLOTINS; UNTERKALMSTEINER; GORSCHEK, 2023).

1.2 Problema

As práticas de engenharia de software aplicadas em uma *startup* são escolhidas de forma a se configurar e se adaptar para promover valor sobre as restrições do contexto da *startup* (SOUZA; MALTA; ALMEIDA, 2017). Segurança e privacidade raramente são priorizadas e posteriormente inseridas ao processo de desenvolvimento, devido à falta de conhecimento, recursos e habilidades dos desenvolvedores, além da pressão exercida pelos clientes e mandatos governamentais (KEKüLLüOğLU; ACAR, 2023). A implementação de requisitos de segurança na maioria dos casos se dá por força de fatores externos, como leis de proteção de dados, ou quando o objetivo da *startup* é o desenvolvimento de software de segurança (KEKüLLüOğLU; ACAR, 2023).

Dessa forma, a priorização pela empresa e a habilidade do time de desenvolvimento na aplicação de práticas de engenharia de software focadas em segurança é importante, pois podem afetar a qualidade do software e a velocidade de desenvolvimento (SOUZA; CICO; MACHADO, 2021). Entretanto, o conhecimento das práticas que trarão melhores resultados dentro do contexto e da fase que a empresa se encontra, não é claro, sobretudo na fase inicial de vida de uma *startup*.

1.3 Objetivos

Nessa Seção são apresentados o objetivo geral e os objetivos específicos deste trabalho, importantes definições que nortearam o desenvolvimento do mesmo e que servem para esclarecer sobre os resultados esperados (GIL, 2017).

1.3.1 Objetivo Geral

Usando como base a estrutura definida por Gil (2017), o objetivo precisa delimitar-se a uma dimensão viável e específica, que abrange o universo de estudo, tempo e lugar.

Sendo assim, o objetivo geral (OG) é **identificar boas práticas de segurança para *startups* em estágio inicial a partir de uma revisão multi-vocal de literatura.**

1.3.2 Objetivos Específicos

A fim de especificar ainda mais o objetivo geral, objetivos específicos foram definidos:

- OE1: Identificar práticas de segurança viáveis de implementação no contexto de *startups* em estágio inicial.
- OE2: Organizar as práticas de segurança por critérios similares usados na engenharia de software moderna.

1.4 Metodologia

A metodologia usada no desdobramento deste trabalho é definida e classificada, e um plano metodológico é construído.

Visto que o objetivo deste trabalho é identificar boas práticas de segurança para *startups* de software, em estágio inicial, a natureza é de caráter aplicada, pois o conhecimento baseado na literatura é para solução de problemas específicos.

Quanto à abordagem, têm-se a qualitativa, devido à condição de identificar boas práticas com base na literatura bibliográfica e na literatura cinza, analisando os dados coletados indutivamente.

Literatura Cinza (LC) é definida por Lefebvre, Manheimer e Glanville (2008) como "literatura não publicada formalmente em fontes como livros e periódicos".

Em relação à tipologia e objetivo, é classificada em descritiva, em acordo com a classificação de Gil (2017), em que a pesquisa busca pela "descrição das características de determinada população ou fenômeno".

A técnica empregada será a Revisão Multi-vocal de Literatura (RML), ou *Multivocal Literature Review* (MLR), utilizando o meio de investigação bibliográfica e de literatura cinza. Uma RML é uma revisão sistemática de literatura (RSL) que envolve tanto LC quanto literatura acadêmica (LF).

A partir da adaptação das diretrizes definidas por Kitchenham, Budgen e Brereton (2016) e por Garousi, Felderer e Mäntylä (2019), a RML será conduzida em três fases: Planejamento da Revisão, Realização da Revisão e Documentação da Revisão. Detalhes sobre cada fase e atividades envolvidas serão descritas no **Capítulo 3 Metodologia.**

1.5 Organização do Trabalho

Este trabalho está elaborado em cinco Capítulos, sendo eles:

- **Capítulo 1 Introdução**: dispõe da contextualização, identificação do problema, definição do objetivo e apresentação da metodologia utilizada.
- **Capítulo 2 Referencial Teórico**: é fundamentado os principais conceitos abordados neste trabalho, a partir da literatura.
- **Capítulo 3 Metodologia**: as fases utilizadas durante a condução da pesquisa são definidas e o protocolo da MLR é criado.
- **Capítulo 4 Resultados**: a condução da revisão é demonstrada e os resultados são apresentados.
- **Capítulo 5 Conclusão**: sintetiza os principais resultados obtidos, destaca a relevância do estudo, exibe suas limitações e propõe trabalhos futuros.

2 Referencial Teórico

2.1 Startups de Software

O termo startup apareceu pela primeira vez na literatura da engenharia de software em 1994, quando Carmel (1994) desenvolveu um estudo exploratório sobre *time-to-completion accelerators* (aceleradores de tempo para conclusão) em startups de pacotes de software. Carmel concluiu que mais estudos eram necessários acerca do processo de desenvolvimento praticado nessas startups, de forma a replicar o sucesso em outros setores de tecnologia (PATERNOSTER *et al.*, 2014).

Startups majoritariamente criam produtos inovadores e de alta tecnologia, buscando a expansão agressiva em mercados altamente escaláveis (PATERNOSTER *et al.*, 2014). Sutton (2000) define características que representam a grande maioria das startups de software, relacionadas a padrões de engenharia e negócio que são moldadas por restrições de operação vivenciadas por esse tipo de empresa:

- **Juventude e Imaturidade** (*Youth and Immaturity*): startups são empresas novas ou relativamente jovens, com pouca experiência e história quando comparadas com organizações maduras e bem estabelecidas, exibindo imaturidade tanto nas capacidades de processo quanto em sua organização.
- **Recursos limitados** (*Limited Resources*): Os recursos são limitados e primordialmente dirigidos a atividades de: lançar e promover o produto e criar alianças estratégicas. "Quanto mais cedo a empresa conseguir realizar essas atividades, maiores serão suas chances de sobrevivência."
- **Múltiplas influências** (*Multiple Influences*): Nos estágios iniciais são particularmente sensíveis a influências de várias fontes: clientes, parceiros, concorrentes, investidores e interna. Divergências entre elas podem ser inconsistentes, por isso essas empresas podem se ajustar e reajustar em como e no que faz.
- **Tecnologias e Mercados Dinâmicos** (*Dynamic Technologies and Markets*): Novas empresas quase sempre começam embarcadas em novidades tecnológicas, criadas para desenvolver produtos inovadores que requerem ferramentas e técnicas de ponta.

O ciclo de vida de uma startup pode ser definido de acordo com o ponto de vista empregado, sempre levando em conta que o aspecto de aprendizagem é crucial para o desenvolvimento de um negócio sustentável (WANG *et al.*, 2016). Ao observar o processo de desenvolvimento do cliente, Wang *et al.* (2016), aborda quatro estágios: definição

ou observação do problema; avaliação do problema; definição da solução; e avaliação da solução.

Considerando o ciclo de vida de uma startup através do processo de desenvolvimento do produto temos duas visões, uma com fases sequenciais e outra paralela. Wang *et al.* (2016) apresenta um ciclo de seis fases paralelas: conceito; em desenvolvimento; protótipo em funcionamento; produto funcional com usuário limitados; produto funcional com alto crescimento; e produto maduro. Crowne (2002), divide a evolução de uma startup em quatro fases sequenciais, do início ao amadurecimento:

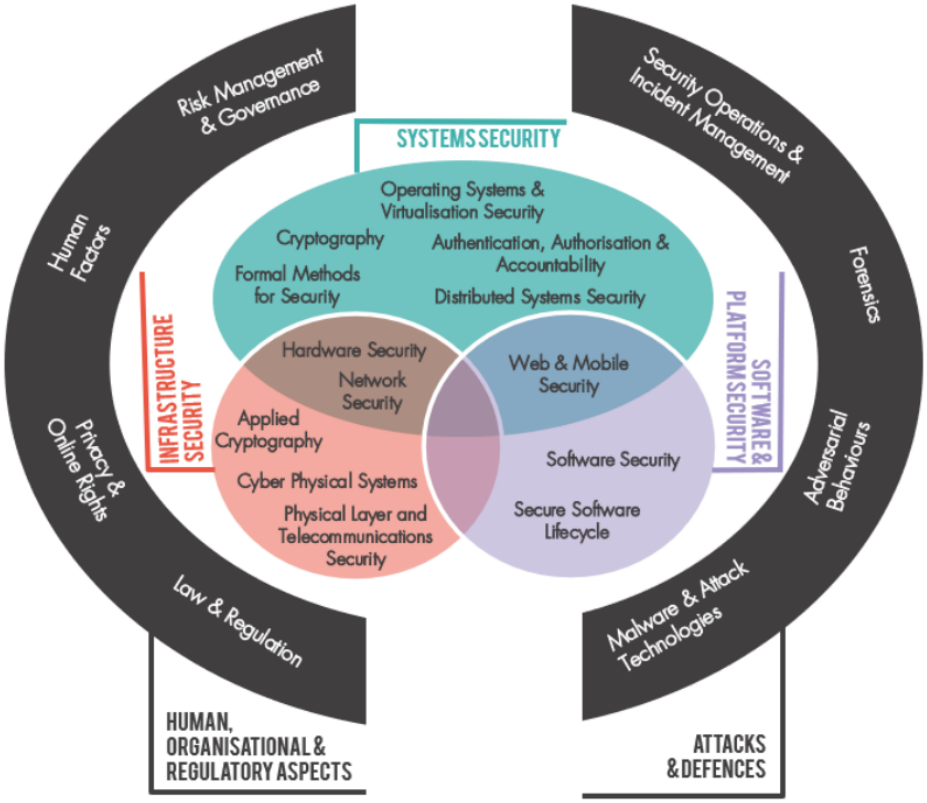
1. **Iniciante** (*Startup*): "Período entre a concepção do produto e a primeira venda". Empreendedores captam uma oportunidade de mercado e sabem como usar a tecnologia para isso, reúnem poucas pessoas com as habilidades necessárias, o dinheiro é escasso, e começam a produzir o produto. Durante essa fase a visão de produto é transformada diretamente da mente dos empresários para linhas de código, a comunicação é rápida e o compromisso e energia são altos (CROWNE, 2002).

2. **Estabilização** (*Stabilization*): "Começa quando o primeiro cliente recebe o produto e termina quando o produto está estável para um novo cliente, sem causar qualquer sobrecarga no desenvolvimento do produto". Novos clientes exigem novas funcionalidades no produto, que pode ainda não estar confiável, o que gera demanda por mais dinheiro e investimento externo. Problemas não resolvidos começam a afetar o crescimento da empresa e se tornam mais complexos de serem sanados, enquanto novos problemas surgem (CROWNE, 2002).

3. **Crescimento** (*Growth*): Começa quando o uso do produto por um novo cliente não causa mais sobrecarga no desenvolvimento do produto. Assim que a participação, taxa de crescimento e tamanho do mercado estiverem estabelecidos, e processos de negócios necessários "para apoiar o desenvolvimento de produtos e as vendas estiverem em vigor" é chegado o fim dessa fase (CROWNE, 2002).

4. **Maturidade** (*Maturity*): "A empresa evoluiu de uma startup para uma organização madura. Tamanho do mercado, participação e taxa de crescimento foram estabelecidos". O processo de desenvolvimento do produto é robusto e previsível, acontece a implementação de novas funcionalidades através da contínua opinião dos clientes e o software abrange normas de qualidade. O produto se encontra estabelecido no mercado de acordo com as necessidades dos clientes e pronto pra migrar para outros mercados (CROWNE, 2002).

Figura 1 – Áreas de conhecimento do escopo CyBOK. Fonte: CyBOK



2.2 Segurança de Software e Plataforma

Segurança de Software e Plataforma é uma ampla categoria que é composta por três áreas de conhecimento de alto nível, sendo elas Segurança de Software, Segurança Web e Mobile e Ciclo de Vida Seguro de Software, conforme definido da Figura 1 (RASHID *et al.*, 2021).

Para o completo entendimento desta Seção, é necessário a definição de alguns termos e conceitos:

- **Software seguro**: A partir do momento que o sistema satisfaz objetivos específicos ou implícitos de segurança pode ser considerado seguro. Os objetivos apresentam requisitos de confidencialidade, disponibilidade e integridade para os dados do sistema e sua funcionalidade (RASHID *et al.*, 2021).
- **Vulnerabilidade de Implementação**: São causas subjacentes (trechos de código) que violam objetivo(s) de segurança, possibilitando ataques, bem como causas (classe de erros) que habilitam técnicas de ataque (RASHID *et al.*, 2021). Bojanova e Galhardo (2023) define vulnerabilidade de segurança como uma cadeia de fraquezas (um defeito, operação e por fim um erro) que leva a uma falha de segurança, "começa com um defeito de código ou especificação e termina com um erro terminal, que, se explorado, leva a uma falha de segurança".

- **Falha de segurança**: Cenário em que o software não satisfaz objetivos de segurança ocasionando em vulnerabilidades, ou quando vulnerabilidades de implementação são exploradas, gerando uma pertubação substancial no comportamento do sistema (RASHID *et al.*, 2021).
- **Práticas de codificação seguras**: Conforme definido por Rashid *et al.* (2021), são "coleções de regras e recomendações que descrevem e ilustram padrões de código bons e ruins".

Para este trabalho, as áreas de conhecimento abordadas serão, respectivamente, Segurança de Software e Ciclo de Vida Seguro de Software.

2.2.1 Segurança de Software

Essa área de conhecimento, segundo Rashid *et al.* (2021), contempla "categorias de vulnerabilidades de implementação de software e de técnicas que podem ser usadas para prevenir ou detectar tais vulnerabilidades, ou para mitigar sua exploração".

2.2.1.1 Categorias de Vulnerabilidades

Vulnerabilidades de implementação são um importante tópico no trabalho da cibersegurança, existem diversas formas de descrição e listagem de vulnerabilidades, sendo comum a concepção que são causadas por práticas inseguras de programação (RASHID *et al.*, 2021). Uma categoria de vulnerabilidades abrange violações a sub-componentes de um sistema de software, sub-sistemas que são definidos com um comportamento esperado e definido pelos desenvolvedores para os clientes, tendo esse comportamento alterado e levado a erros como objetivo pelo autor da violação (RASHID *et al.*, 2021).

Apesar de um software possuir vulnerabilidades de implementação, pode ser possível que estas não resultem numa quebra de um objetivo de segurança, assim como sistemas que não possuem vulnerabilidades de implementação podem falhar em alcançar objetivos de segurança (RASHID *et al.*, 2021). Com isso, é necessário que durante a fase de *design* de um sistema especificações sejam criadas a partir de objetivos de segurança, para que os sub-sistemas possam se comportar a fim de alcançá-las e que as vulnerabilidades de implementação sejam consideradas como violação dessas especificações (RASHID *et al.*, 2021).

1. **Vulnerabilidades de Gerenciamento de Memória**: Acontecem geralmente em linguagens de programação que delegam o gerenciamento de memória ao desenvolvedor, sendo este responsável pela alocação, desalocação e acesso à memória (RASHID *et al.*, 2021). Quando não seguido a definição proposta para o gerenciamento de memória, vulnerabilidades como *spatial vulnerability* e *temporal vulnerability* são fontes

conhecidas de erros, sendo respectivamente definidas como: acesso de *buffer* fora do limite de índice definido; e acesso de memória desalocada que anteriormente foi alocada para o programa (RASHID *et al.*, 2021).

Com o comportamento não definido de uma vulnerabilidade de gerenciamento de memória presente nas linguagens como C e C++, corrupção de: dados do programa, fluxo de controle do programa e código do programa podem acontecer devido ao acesso e mutação da memória destinada ao código compilado e pilha de chamadas (RASHID *et al.*, 2021). As classes de ataque responsáveis por essas corrupções podem ser definidas como: *code corruption attack*, *control-flow hijack attack*, *data-only attack* e *information leak attack* (RASHID *et al.*, 2021).

2. **Vulnerabilidades de Geração de Resultados Estruturados**: Frequentemente programas são criados para possuir *outputs* (resultados) estruturados para serem consumidos por outro programa, em que de acordo com determinado conjunto de *inputs* (parâmetros de entrada) é esperado determinado comportamento e produto resultante (RASHID *et al.*, 2021). Este tipo de vulnerabilidade utiliza *inputs* modificados para gerar saídas estruturadas não intencionais, devido ao uso da prática insegura de programação de utilizar dados provenientes do *input* (RASHID *et al.*, 2021).

   Essas vulnerabilidades são conhecidas como *injection vulnerabilities*, dentre elas as mais conhecidas são: SQL *injection*, *command injection*, *script injection*, XPath *injection*, HTML *injections* e CSS *injection* (RASHID *et al.*, 2021). As técnicas de ataque dependem da linguagem utilizada no resultado estruturado (RASHID *et al.*, 2021).

3. **Vulnerabilidades de Condição de Corrida**: Um programa assume suposições de como outros programas, *threads* ou processos (atores) irão se comportar e agir com a memória compartilhada por eles, sendo que as suposições fazem parte do contrato que especifica como o ambiente de execução irá trabalhar com os recursos do programa (RASHID *et al.*, 2021). Violações nessa especificação são erros de concorrência, conhecidos também como corrida de condições, que segundo Rashid *et al.* (2021), tem como consequência que o "comportamento do programa pode depender de qual ator simultâneo acessa um recurso primeiro ("ganha uma corrida")".

   Erros de concorrência também geram vulnerabilidades de segurança, introduzindo o não determinismo no sistema, onde o invasor com controle do tempo de ações dos atores pode ter controle no comportamento do ambiente, afetando objetivos de segurança (RASHID *et al.*, 2021). Segundo Rashid *et al.* (2021), duas grandes áreas onde ocorrem vulnerabilidades de condição de corrida são no sistema de arquivos e no estado de sessão de aplicações web.

4. **Vulnerabilidades de API**: *Application Programming Interface* (API) são componentes de software utilizados como interface para comunicação com outros componentes, como bibliotecas, serviços, serviços web e sistemas, onde a violação de contrato de comportamento entre ambos gera vulnerabilidades de segurança (RASHID *et al.*, 2021).

2.2.1.2 Prevenção de Vulnerabilidades

Violações no contrato de especificação propostas pela linguagem ou API promovem, como visto na Seção anterior, vulnerabilidades de implementação, por isso conforme Rashid *et al.* (2021) explica, as melhores abordagens erradicam categorias de vulnerabilidades através do *design* (desenho) da linguagem ou API, e, quando não possível o *re-design* (redesenho), o uso de práticas de codificação seguras.

Adoção de práticas de codificação segura, segundo OWASP (2022), comumente resulta no barateamento da construção de software, uma vez que a correção de problemas e a ocorrência de falhas podem elevar o custo. Essas práticas são adotadas ao ciclo de desenvolvimento de software, a partir da enumeração de requisitos de segurança e casos de abuso nas fases iniciais do ciclo (OWASP, 2022). OWASP (2022) define casos de abuso: "descrevem o mau uso, intencional ou não, do software" desafiando "os pressupostos do projeto do sistema".

Linguagens podem prevenir vulnerabilidades de gerenciamento de memória e de outras categorias, através do controle, alocação e desalocação de memória, combinando checagens dinâmicas e estáticas e procurando evitar: mutação de estados, alocação dinâmica e desalocação estática de memória com o uso de *garbage collection* (RASHID *et al.*, 2021). Para as que delegam o controle para o desenvolvedor, é possível criar implementações que controlem de forma segura a memória, mas ao custo da performance (RASHID *et al.*, 2021).

As regras e recomendações de codificação segura são criadas a partir da experiência de programadores, que sumarizam práticas específicas de certas linguagens ou práticas comuns que podem ser aplicadas a diversas linguagens (RASHID *et al.*, 2021). Conforme a linguagem evolui o número de recomendações pode crescer, e é provável que os desenvolvedores se afastem da aplicação delas, por isso Rashid *et al.* (2021) manifesta a importância de "fornecer suporte de ferramentas para verificar a conformidade do software com as regras de codificação".

As práticas de codificação segura expostas no guia por OWASP (2022) oferecem cada uma lista de verificações que facilitam aos desenvolvedores e especialistas de segurança a implementação destas no projeto em desenvolvimento, ajudando na prevenção de vulnerabilidades.

2.2.1.3 Detecção de Vulnerabilidades

Durante a fase de desenvolvimento, teste ou manutenção de um software, técnicas para detecção de vulnerabilidades podem ser aplicadas, em adicional, ou na não possibilidade de aplicação, das técnicas utilizadas para prevenção de vulnerabilidades (RASHID *et al.*, 2021). Técnicas como detecção estática e dinâmica, revisão de código manual e auditoria são amplamente indicadas e estudadas na área da ciência da computação (RASHID *et al.*, 2021).

A detecção estática é feita a partir da análise do código fonte ou binário procurando por quebras de especificação de contrato de linguagens ou APIs, na tentativa de prever todas as execuções possíveis do programa, com a vantagem de não precisar que o código seja executável (RASHID *et al.*, 2021). Segundo Rashid *et al.* (2021), duas classes importantes de técnicas são a **detecção estática heurística** e a **verificação estática robusta**, onde as técnicas buscam, respectivamente:

- Detectar e reportar violações a regras de codificação de heurísticas de práticas de programação segura formais;
- Concluir que determinado programa não possui nenhuma vulnerabilidade pertencente a uma categoria de vulnerabilidades.

Detecção dinâmica consiste no monitoramento da execução de um programa em busca de vulnerabilidades, sendo sua aplicação dependente dos aspectos relacionados a categoria de vulnerabilidades alvo do monitoramento e de quais execuções monitorar (RASHID *et al.*, 2021). Com o objetivo de gerar execuções suficientes para cobrir falhas de um programa, testes de software utilizam **técnicas de caixa branca e caixa preta** para desenvolvimento de entradas apropriadas, que respectivamente:

- Analisam a estrutura interna do programa criando entradas que executam diferentes caminhos de condições (RASHID *et al.*, 2021).
- Analisam o comportamento do programa criando entradas apropriadas de acordo com seu domínio de valor, sem levar em conta a estrutura interna do programa (RASHID *et al.*, 2021).

2.2.1.4 Mitigação de Exploração de Vulnerabilidades

Técnicas para mitigação de exploração de vulnerabilidades são importantes para limitar o impacto na performance de programas sob ataque, ampliar a compatibilidade com programas legados e complementar a aplicação de técnicas de prevenção e detecção (RASHID *et al.*, 2021). São aplicadas geralmente na infraestrutura de execução do

programa, tendo como técnicas mais importantes, elencadas por Rashid *et al.* (2021): detecção de ataques em tempo de execução, diversificação de software automática, limitação de privilégios e checagens de integridade de software.

2.2.2 Ciclo de Vida de Software Seguro

Um ciclo de vida de software seguro engloba componentes abrangentes de desenvolvimento de software, a fim de detectar, evitar e corrigir defeitos de segurança, tal como responder em caso de exploração, durante todo o processo de desenvolvimento, desde o *design* até sua manutenção em ambiente de produção (RASHID *et al.*, 2021). Esta Seção apresentará dois processos prescritivos de ciclo de vida de software seguro, devido às suas características de contemplação de diversas fases do ciclo de vida de software e integração entre processos.

2.2.2.1 Microsoft *Security Development Lifecycle* (SDL)

Esse processo foi criado pela Microsoft (2023) com o objetivo de prescrever práticas de desenvolvimento que dão suporte à garantia de requisitos de segurança e de conformidade, ajudando desenvolvedores a escrever software seguro, diminuindo a gravidade de vulnerabilidades, com custo menor (MICROSOFT, 2023).

É descrito em 12 práticas, conforme listado abaixo:

1. **Fornecimento de Treinamento (*Provide Training*)**: Desenvolvedores, gerentes, engenheiros e outros devem estar cientes da importância da segurança de software, assim como o produto deve acompanhar as necessidades de negócio (MICROSOFT, 2023). Dessa forma, treinamentos quanto ao impacto, objetivos e formas de ataque devem ser apresentadas para toda a equipe, assim como padrões, requisitos, práticas e novas técnicas de segurança (MICROSOFT, 2023).

2. **Definir Requisitos de Segurança (*Define Security Requirements*)** Requisitos de segurança e privacidade devem ser definidos nas fases iniciais de *design* do software, a partir de padrões de segurança, experiência com incidentes passados, práticas de desenvolvimento e ameaças modeladas (MICROSOFT, 2023). Mas sua atualização durante o ciclo de vida do software é importante também, de acordo com mudanças na funcionalidades e ameaças existentes (MICROSOFT, 2023). Rashid *et al.* (2021) apresenta o *Security Quality Requirements Engineering* (SQUARE) como uma técnica de nove passos usada nos estágios iniciais de desenvolvimento para garantir segurança.

3. **Definir Métricas e Relatório de Conformidade (*Define Metrics and Compliance Reporting*)**: A definição de níveis mínimos de qualidade de segurança são

importantes para a definição de metas para o time e para criação de métricas de segurança, que ajudam no entendimento de riscos associados a problemas de segurança, resolução de defeitos durante o desenvolvimento e categorização de vulnerabilidades de segurança (MICROSOFT, 2023). Essas métricas geram Indicadores Chave de Performance (*Key Performance Indicators* (KPI)) importantes para relatórios de gestão (RASHID *et al.*, 2021).

4. **Executar Modelagem de Ameaças (*Perform Threat Modeling*)**: Modelagem de ameaças acontece quando o time considera, documenta e discute as implicações de segurança presentes no ambiente operacional planejado, levando em conta pontos fortes e fracos do sistema ao se defender em diferentes cenários de ameaças, de forma estruturada (RASHID *et al.*, 2021). Rashid *et al.* (2021) apresenta uma abordagem que consiste em: (1) atores maliciosos e benevolentes do sistema; (2) *design* dos componentes e sistema; (3) limites de confiança do sistema; (4) fluxo de dados dentro e fora dos limites de confiança através dos atores do sistema.

5. **Estabelecer Requisitos de *Design* (*Establish Design Requirements*)**: Requisitos de *Design* definem recursos de segurança a serem implementados no software, como autenticação, criptografia e *logging*, mas o seu estabelecimento têm-se mostrado complicado e resultado em implementações e escolhas que levam a vulnerabilidades (MICROSOFT, 2023). Rashid *et al.* (2021) apresenta uma técnica criada por Saltzer e Schroeder que consiste de oito princípios para o design de funcionalidades seguras: *Economy of mechanism; Fail-safe defaults; Complete mediation; Open design; Separation of privilege; Least privilege; Least commom mechanism; Psychological acceptability.*

6. **Definir e Usar Padrões de Criptografia (*Define and Use Cryptography Standards*)**: Criptografia é importante para prevenir que os dados sensíveis transmitidos e armazenados por um sistema sejam alterados ou divulgados, garantindo segurança e privacidade de dados (MICROSOFT, 2023). Microsoft (2023) alerta sobre a incorreta implementação de padrões de criptografia que podem levar a falhas e brechas na segurança, indicando a consulta por especialistas no assunto e ao uso de bibliotecas de criptografias com baixo acoplamento no código que possam ser facilmente atualizadas ou substituídas.

7. **Gerenciar Risco de Segurança do Uso de Componentes de Terceiros (*Manage the Security Risk of Using Third-Party Components*)**: Componentes de terceiros e *open source* são largamente utilizados atualmente, nos mais variados tipos de projetos, e sua escolha e uso devem ser respaldados na segurança e potencial inserção de vulnerabilidades no sistema a ser integrado (MICROSOFT, 2023).

Manter uma lista de componentes confiáveis e verificados é importante, assim como maneiras de lidar com vulnerabilidades descobertas nesses (RASHID *et al.*, 2021).

8. **Usar Ferramentas Aprovadas (Use Approved Tools)**: Microsoft (2023) descreve que manter uma lista de ferramentas aprovadas e vinculadas a verificações de segurança é importante, sendo os engenheiros responsáveis pelo esforço de utilizar versões atualizadas assim como manter compatibilidade com versões anteriores.

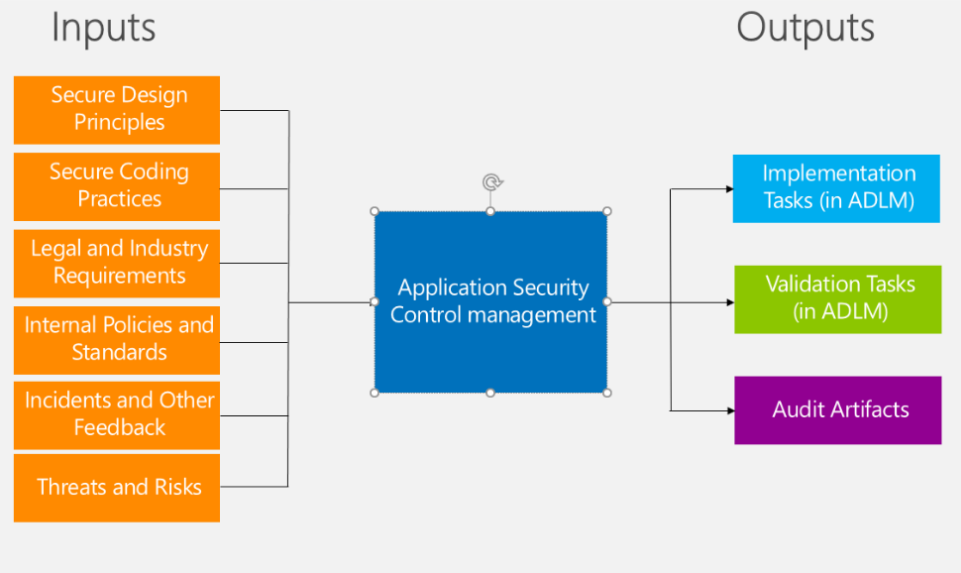
9. **Executar Testes de Segurança de Análise Estática (*Perform Static Analysis Security Testing* (SAST))**: Ferramentas de análise estática são importantes para a verificação de conformidade com contrato da linguagem e padrões de código, conforme mostrado na Seção 2.2.1.3. Seu uso é atribuído em *pipelines* de integração contínua (*Continuos Integration* (CI)), funcionando como pré-requisito para integração, e para maior eficiência, no ambiente de desenvolvimento dos programadores (RASHID *et al.*, 2021).

10. **Executar Testes de Segurança de Análise Dinâmica (*Perform Dynamic Analysis Security Testing* (DAST))**: Esses testes são importantes pois executam verificações de segurança em tempo de execução de software integrado ou empacotado, conforme exposto na Seção 2.2.1.3. Assim como os testes de segurança de análise estática, são usados em CI e ambiente de desenvolvimento, além de integrados também em *pipelines* de entrega contínua (*Continuos Delivery* (CD)) (RASHID *et al.*, 2021).

11. **Executar Testes de Penetração (*Perform Penetration Testing*)** Testes de penetração são análises de segurança de sistema executadas por "profissionais de segurança qualificados simulando as ações de um hacker" (MICROSOFT, 2023). Rashid *et al.* (2021) conceitua como testes de caixa preta, descrito na Seção 2.2.1.3, que buscam descobrir qualquer vulnerabilidade oriundas de erros de implementação, falhas no *design*, entrega e sistema e etc. Rashid *et al.* (2021) cita um importante recurso para a estruturação de testes de penetração, o Projeto de Segurança de Aplicativos Web Aberto (*Open Web Application Security Project* (OWASP)) Top 10 *Most Critical Web Application Security Risks*.

12. **Estabelecer Processo Padrão de Resposta a Incidentes (*Establish a Standard Incident Response Process*)**: O estabelecimento desse processo pela empresa é importante para uma rápida e organizada resposta em ocorrência de ataques inevitáveis, que inclui: quem contatar em caso de emergência de segurança, estabelecimento de um protocolo de mitigação de vulnerabilidade, comunicação e resposta com cliente e plano de rápida entrega de correção (RASHID *et al.*, 2021). O processo deve ser incrementado a cada uso com novos aprendizados e testado antes de ser requisitado (RASHID *et al.*, 2021).

Figura 2 – Gerenciamento de Controle de Segurança de Aplicativos. Fonte: SAFECode (SAFECODE, 2018)



2.2.2.2 Software Assurance Forum for Excellence in Code (SAFECode)

A organização SAFECode publicou um guia chamado "*Fundamental practices for secure software development: Essential elements of a secure development lifecycle program*" que contêm práticas fundamentais para para desenvolvimento seguro de software, sendo essas práticas formas de garantir a capacidade do software de resistir à ataques (RASHID *et al.*, 2021).

As oito práticas fundamentais serão descritas resumidamente abaixo:

1. **Definição de Controle de Segurança de Aplicativos (*Application Security Control Definition*)**: SAFECode utiliza o termo Controles de Segurança de Aplicativos (*Application Security Controls* (ASC)) para se referir aos requisitos de segurança, se assemelhando a prática dois descrita no processo SDL. A Figura 2 apresenta as entradas e saídas do controle.

2. ***Design***: A partir dos requisitos de segurança estabelecidos, é necessário incorporar funcionalidades de segurança, e a arquitetura e *design* devem estar preparadas para isso (SAFECODE, 2018). O guia SAFECode (2018) apresenta práticas e funcionalidades essenciais para produzir sistemas protegidos contra ataques de qualquer causa, dentre elas: modelagem de ameaças; execução de revisão arquitetural e de *design*; desenvolvimento de estratégia de criptografia; padronização de identidade e gerenciamento de acesso; utilização de *logging*; práticas de auditoria.

3. **Práticas de Codificação Seguras (*Secure Coding Practices*)**: Erros podem ser introduzidos no programa quando desenvolvedores escrevem o código, levando

a vulnerabilidades que comprometem o software (SAFECODE, 2018). Minimizar esses erros de nível de código faz parte do objetivo de segurança, e pode ser alcançado: definindo padrões de codificação; selecionando linguagens, *frameworks* e bibliotecas apropriadas; utilizando ferramentas de análise automática; revisando o código manualmente (SAFECODE, 2018).

4. **Gerenciar Riscos de Segurança Inerentes ao Uso de Componentes de Terceiros (*Manage Security Risk Inherent in the Use of Third-Party Components*)**: Prática muito semelhante à abordada pela Microsoft na sétima prática do processo SDL.

5. **Teste e Validação (*Testing and Validation*)**: Prática muito semelhante a abordada pela Microsoft na décima e décima primeira prática do processo SDL.

6. **Gerenciar Descobertas de Segurança (*Manage Security Findings*)**: Os achados pelas primeiras cinco práticas do SAFECode resultam em artefatos relacionados a segurança do produto (ou falta dela), seu rastreamento deve ser realizado a fim de tomar ações para remediar, mitigar ou aceitar o risco respectivo (SAFECODE, 2018). O rastreio deve ficar disponível para todo o time, tendo como requisitos: definição de problemas de segurança, definição de graus de gravidade, riscos residuais provenientes da mitigação de problemas, priorização de problemas de acordo com gravidade, criação de sistema de monitoramento (SAFECODE, 2018).

7. **Resposta a Vulnerabilidades e Divulgação (*Vulnerability Response and Disclosure*)**: Prática muito semelhante a abordada pela Microsoft na décima segunda prática do processo SDL.

8. **Planejando a Implementação e Implantação de Desenvolvimento Seguro (*Planning the Implementation and Deployment of Secure Development*)**: De acordo com SAFECode (2018), um ciclo de vida maduro e saudável inclui as sete práticas acima e a integração dessas práticas dentro do "processo de negócio e toda a organização, incluindo gerenciamento de programa, *stakeholders*, planejamento de entrega, métricas e indicadores e plano para melhoria continua".

2.3 Trabalhos Relacionados

Lanka, Varol e Shashidhar (2023) definem um modelo para auxiliar startups de software, com foco em desenvolvimento de SaaS (*Software as a Service*), a trabalhar com segurança e permitir que serviços baseados em confiança possam ser usados como um fator de diferenciação. A abordagem utilizada foi de identificação de abordagens de implementação de segurança com pouco capital e recursos, e avaliação através de validação ontológica de padrões de conformidade da indústria. Por fim, o modelo definido exibe

microarquiteturas exclusivas para controle de segurança ou conformidade, com baixo custo de implementação, que ajudam startups "a permanecerem seguras e atingir metas de conformidade"(LANKA; VAROL; SHASHIDHAR, 2023).

Através de um estudo de caso aplicado a quatro startups incubadas em universidades, Souza, Malta e Almeida (2017), geraram o modelo ASM (*Academic Startup Model*) que revela sete principais características de alto nível, relacionadas a aplicação de práticas de engenharia de software em startups incubadas. As características giram em torno do rápido desenvolvimento e possuem como causa-efeito: falta de recursos; baixa prioridade em aspectos de qualidade; time de desenvolvedor catalisador; acúmulo de débito técnico; crescimento inicial esconde desempenho; e abordagem evolucional. Em conclusão, foi comparado o modelo ASM com o GSM (*Greenfield Startup Model*), que aborda prioridades de startups, e as mesmas características foram encontradas, demonstrando que startups em estágio inicial não aplicam ou quase não aplicam práticas de engenharia de software, majoritariamente práticas de segurança.

Através de métodos mistos, design através de entrevistas e workshops com startups dinamarquesas e especialistas em segurança cibernética, Li *et al.* (2023) avaliaram a concepção e adoção de segurança em startups de tecnologia de saúde. O estudo mostrou que embora tida como essencial a implementação de segurança pelas startups, é muitas vezes imprecisa e incerta sua prioridade diante de tantas prioridades da empresa, e o aperfeiçoamento e treinamento em segurança é autodidata pelos funcionários (LI *et al.*, 2023).

3 Metodologia

Este capítulo apresenta a metodologia e os procedimentos utilizados para a identificação e organização de práticas de segurança para *startups* a partir de uma revisão multi-vocal da literatura (RML).

3.1 Diretrizes

Como engenharia de software é um campo orientado a prática e aplicação, com tópicos de pesquisa derivados da indústria de software, o reconhecimento da LC deve ser formalizado devido a sua capacidade de poder identificar tendências emergentes nessa mesma indústria (GAROUSI; FELDERER; MäNTYLä, 2019). Por isso, a RML diferentemente da RSL, adota como evidência não só literatura formal publicada e revisada (LF), mas também a LC, onde a ideia geral é captar diferentes pontos de vista de um tópico contemporâneo de diversos autores envolvidos, divulgados em várias formas de escrita, incorporando e enriquecendo a pesquisa (GAROUSI; FELDERER; MäNTYLä, 2019).

Garousi, Felderer e Mäntylä (2019) propuseram diretrizes para a condução de uma RML em engenharia de software a partir da sintetização de regras advindas de outras áreas de conhecimento e da experiência em construir RML. Esse tipo de revisão é bastante comum em outras áreas de conhecimento, unindo o estado da arte e a visão dos praticantes em uma área específica (GAROUSI; FELDERER; MäNTYLä, 2019).

Para este trabalho foi ajustada a estrutura padrão de criação de um protocolo de RSL por Kitchenham, Budgen e Brereton (2016) com adaptações conforme diretrizes de uma RML por Garousi, Felderer e Mäntylä (2019). Consiste em três fases principais, as quais são sínteses de atividades específicas, Fase 1 - Planejamento da RML, Fase 2 - Condução da RML e Fase 3 - Documentação da RML:

- Fase 1 - Planejamento da Revisão
    1. Desenvolvimento do Protocolo
- Fase 2 - Condução da Revisão
    2. Processo de Busca
    3. Seleção dos estudos
    4. Avaliação da qualidade
    5. Extração dos dados

6. Síntese

- Fase 3 - Documentação da Revisão

7. Apresentação da análise dos dados obtidos

3.2 Planejamento da Revisão

Dentro da Fase 1, temos como produto gerado o protocolo de revisão sistemática, que desempenha papel importante, fornecendo uma estrutura que possibilita a tomada de decisões acerca do desenho do estudo (KITCHENHAM; BUDGEN; BRERETON, 2016).

O protocolo tenta descrever de maneira mais detalhada possível como a revisão será conduzida (KITCHENHAM; BUDGEN; BRERETON, 2016). Um protocolo bem produzido permite rápidas alterações durante o andamento da revisão, além de guiar o realizador do estudo (KITCHENHAM; BUDGEN; BRERETON, 2016). O Protocolo gerado é dividido nos seguintes componentes e Seções:

1. Necessidade de uma RML, Seção 3.2.1;
2. Questões de Pesquisa, Seção 3.2.2;
3. Estratégia de Busca, Seção 3.2.3;
4. Seleção de Estudos, Seção 3.2.4;
5. Avaliação de Qualidade, Seção 3.2.5;
6. Extração dos Dados, Seção 3.2.6;
7. Síntese, Seção 3.2.7.

3.2.1 Necessidade de uma RML

Antes de iniciar o desenvolvimento do protocolo é necessário entender o porquê da escolha executar uma RML ao invés de uma RSL ou outros estudos de mapeamento. Garousi, Felderer e Mäntylä (2019) sugere questões que podem ser levadas em conta no momento dessa decisão, conforme Quadro 1, em que um ou mais "sim" justificam a inclusão de LC, e quanto mais respostas positivas maior é a sugestão de adoção de uma RML.

Quadro 1 – Questões para decidir quando adotar LC na revisão adaptado de Garousi, Felderer e Mäntylä (2019)

ID	Questão	Resposta	Justificativa
1	O assunto é "complexo" e não pode ser resolvido considerando apenas a literatura formal?	Sim	Devido ao caráter extremamente contextual do objeto de pesquisa a complexidade considerada aqui é devido a não existência de abundante evidência na literatura formal
2	Há falta de volume ou qualidade de evidências, ou falta de consenso sobre medição de resultados na literatura formal?	Sim	Durante a investigação do tema pouco volume foi encontrado
3	As informações contextuais são importantes para o assunto em estudo?	Sim	Extremamente importantes, devido ao foco em *startups* em estágio inicial
4	O objetivo é validar ou corroborar resultados científicos com experiências práticas?	Sim	Existem diversas práticas de segurança sugeridas na literatura, e a que ponto cabem ser aplicadas a *startups*
5	O objetivo é desafiar suposições ou falsificar resultados da prática usando pesquisas acadêmicas ou vice-versa?	Não	
6	Uma síntese de *insights* e evidências da comunidade industrial e acadêmica seria útil para uma ou ambas as comunidades?	Sim	Contexto, dificuldades, abordagens e necessidades da comunidade industrial acerca da prática de segurança de software enriquece o estudo do tema
7	Existe um grande volume de fontes de profissionais indicando alto interesse dos profissionais em um tópico?	Sim	

3.2.2 Questões de Pesquisa

As questões da pesquisa guiam todo o processo da revisão, promovendo uma base para decisões relacionadas a: estratégia de busca; quais estudos primários incluir; quais dados devem ser extraídos; como sintetizá-los para responder às questões (KITCHENHAM; BUDGEN; BRERETON, 2016).

As questões de pesquisa levantadas são:

- QP1 - Quais as práticas de segurança de software tem sido utilizada por *startups*?
- QP2 - Quais práticas de segurança de software são mais adaptáveis para *startups* em estágio inicial?
- QP3 - Quais as principais dificuldades enfrentadas pelas *startups* para aplicar as práticas de segurança?

3.2.3 Estratégia de Busca

Para encontrar conteúdo relevante que possa responder as questões de pesquisa é necessário um conjunto de técnicas de busca, dado que LC e LF são publicados e/ou postados em fontes e formatos diferentes. Os métodos de busca escolhidos foram:

- **Busca Automatizada**: Este método usa como recurso bibliotecas digitais, serviços de indexação e motores de busca, em que são aplicadas *strings* de busca derivadas do:

contexto da revisão, conhecimento do autor, questões de pesquisa e palavras-chave extraídas do conjunto ouro (KITCHENHAM; BUDGEN; BRERETON, 2016).

  - Para LF: A *string* de busca será aplicada nas bibliotecas digitais e serviços de indexação.
  - Para LC: A *string* de busca será aplicada em motor de busca.

- **Busca Manual por Conferência**: A partir de uma Conferência com enfoque na área de estudo, uma busca manual é realizada nos artigos publicados na mesma de forma a rastrear estudos adicionais. Essa estratégia se assemelha ao *Backwards Snowballing*, em que buscas são realizadas a partir de citações presentes em publicações candidatas a fim de encontrar estudos adicionais (KITCHENHAM; BUDGEN; BRERETON, 2016).

3.2.3.1 Bases

As bases (bibliotecas digitais) escolhidas foram: ACM Digital Library, IEEE Digital Library e Scopus.

O motor de busca escolhido para LC foi o Google.

3.2.3.2 Definição de Parada

Para a busca automatizada em bibliotecas digitais o critério de parada será a exaustão de dados, ou seja, todas as evidências retornadas estarão aptas para a seleção.

Para a busca automatizada em motor de busca o critério será diferente, devido ao grande número de resultados que podem ser retornados, ficando assim definido pelo conjunto dos critérios limite de esforço e saturação teórica. Ambos são apresentados por Garousi, Felderer e Mäntylä (2019): o limite de esforço restringe a coleta de evidências aos N primeiros resultados; a saturação teórica é um critério subjetivo relacionado ao declínio da qualidade e surgimento de novos conceitos conforme a busca continua. Sendo assim, para esta busca, as primeiras 70 evidências serão coletadas e, posteriormente, encerrará se atingida a saturação teórica.

3.2.3.3 Conjunto Ouro

O Conjunto Ouro, exposto no Quadro 2, apresenta três estudos relevantes para o tópico de pesquisa, encontrados a partir de busca manual nas bases digitais. Eles serão importantes na definição das palavras-chave e na avaliação do resultado da busca automatizada.

Quadro 2 – Estudos do Conjunto Ouro

Base	**Título**	**Citação**
IEEE	Strategies for a Startup Software-as-a-Service Organizations with Minimal Budget to Achieve Security and Compliance Goals	(LANKA; VAROL; SHASHIDHAR, 2023)
IEEE	Software Engineering in Startups: A Single Embedded Case Study	(SOUZA; MALTA; ALMEIDA, 2017)
CIBSE	A Survey on Software Engineering Practices in Brazilian Startups	(SOUZA; CICO; MACHADO, 2021)

3.2.3.4 PICOC

A estratégia PICOC (*Population, Intervention, Comparison, Outcome e Context*) é um *framework* que ajuda na definição do escopo da pesquisa (WOHLIN, 2012). A partir dessa estratégia, os seguintes termos foram determinados no Quadro 3.

Quadro 3 – Aplicação do *framework* PICOC

Conceito	**Descrição**	**Termos**
População (Population)	Qual a população/área de aplicação em que serão coletadas as evidências	Startups
Intervenção (Intervention)	Metodologia, procedimentos, tecnologia sobre estudo	Práticas de Segurança de Software
Comparação (Comparision)	Metodologia, procedimentos, tecnologia que está sendo usada em comparação	Empresas consolidadas (maduras)
Resultado (Outcome)	Após aplicação da intervenção, resultados relevantes observados	Software seguro
Contexto (Context)	Local de aplicação da intervenção	Startups em estágio inicial

3.2.3.5 Palavras-chave e Sinônimos

As palavras-chave foram escolhidas a partir do conjunto ouro (presente no Quadro 2) e dos termos definidos no PICOC (apresentado no Quadro 3), e estão consolidadas na Tabela 1. A coluna “Relacionado a” define o relacionamento entre a palavra-chave e um conceito PICOC.

Conforme será exposto na próxima Seção, a aplicação da primeira versão da *string* de busca, derivada das palavras-chave expostas na Tabela 1, levantou a necessidade de re-definição das palavras-chave, ao passo que não representavam a intervenção e os resultados desejados.

Tabela 1 – Primeira versão do conjunto de palavras-chave

ID	Palavra(s)-chave	Sinônimos	Relacionado a
1	Agile Software Development		Intervenção
2	Development Practices		Intervenção
3	Secure Software Development		Intervenção
4	Security		Intervenção
5	Software Engineering Practices		Intervenção
6	Software Security Practices	Software Security Approaches	Intervenção
7	Startups		População
8	Strategies		Intervenção
9	Secure Software		Resultado

Por isso, para de forma abrangente englobar as áreas focais da revisão, as palavras-chave foram redefinidas e uma nova versão criada, presentes na Tabela 2.

Tabela 2 – Versão final do conjunto de palavras-chave

ID	Palavra(s)-chave	Sinônimos	Relacionado a
1	Startup		População
2	Security		Intervenção
3	Vulnerability		Intervenção
4	Secure		Resultado
5	Software		Contexto

3.2.3.6 *String* de Busca

Temos a primeira versão da *string* de busca gerada, a partir da primeira versão das palavras-chave da Tabela 2:

- ("Startups"OR "startup") AND ("Agile Software Development"OR "Development Practices"OR "Secure Software Development"OR "Security"OR "Software Engine-

ering Practices"OR "Software Security Practices"OR "Software Security Approaches"OR "Strategies"OR "Secure Software").

A *string* de busca acima foi aplicada nas bases selecionadas, diferenciando quanto à sintaxe de acordo com as regras de busca avançada exigidas, conforme é exposto na Tabela 11 presente no Apêndice A.

Após análise da performance da busca automatizada realizada com a primeira versão da *string* de busca, foram encontrados estudos que fugiam do escopo conhecido e denotado para este trabalho; por isso, a partir da versão final de palavras-chave (Tabela 2) e da recomendação de Kitchenham, Budgen e Brereton (2016) de manter *strings* de busca simples, uma nova versão foi gerada:

- Startup AND (Security OR Secure OR Vulnerability) AND Software.

Na Tabela 3 são apresentadas as variações construídas para cada base, assim como o número de resultados encontrados.

Tabela 3 – Versão final da *String* de busca

Base / Motor de Busca	**String**	**Resultados**
IEEE Xplore	("Document Title":Startup AND ("Document Title":Security OR "Document Title":Secure OR "Document Title":Vulnerability) AND "Document Title":Software) OR ("Abstract":Startup AND ("Abstract":Security OR "Abstract":Secure OR "Abstract":Vulnerability) AND "Abstract":Software) OR ("Author Keywords":Startup AND ("Author Keywords":Security OR "Author Keywords":Secure OR "Author Keywords":Vulnerability) AND "Author Keywords":Software) OR ("Index Terms":Startup AND ("Index Terms":Security OR "Index Terms":Secure OR "Index Terms":Vulnerability) AND "Index Terms":Software)	41
ACM	Title:(("startup") AND ("Security"OR "secure"OR "vulnerability") AND ("software")) OR Abstract:(("startup") AND ("Security"OR "secure"OR "vulnerability") AND ("software")) OR Keyword:(("startup") AND ("Security"OR "secure"OR "vulnerability") AND ("software"))	9
SCOPUS	ABS(startup AND (security OR secure OR vulnerability) AND software) OR TITLE(startup AND (security OR secure OR vulnerability) AND software) OR KEY (startup AND (security OR secure OR vulnerability) AND software)	135
Google	allintext: Startup AND AND Software Security OR Secure OR Vulnerability	80.100.000

3.2.3.7 Busca Manual por Conferência

De forma a complementar os estudos encontrados pela busca automatizada, foi feita uma exploração manual na conferência "2017 IEEE/ACM 1st International Workshop on Software Engineering for Startups (SoftStart)", exposta na base IEEE Xplore, em

busca de citações que pudessem conter estudos adicionais. Foram encontrados seis artigos ao todo, que, após avaliação do título e resumo, resultaram em dois artigos adicionais à revisão.

3.2.4 Seleção de Estudos

Critérios de inclusão e exclusão são importantes para filtrar estudos que podem contribuir para a formulação de resposta às questões de pesquisa, possuindo assunto relevante para a pesquisa (KITCHENHAM; BUDGEN; BRERETON, 2016). A aplicação dos critérios é feita por etapas, onde critérios mais simples e abrangentes são aplicados primeiramente, e, caso exista dúvida na inclusão do estudo, ele pode ser incluído para que seja avaliado em critérios posteriores ou descartado em fases seguintes (KITCHENHAM; BUDGEN; BRERETON, 2016).

Foram elencados critérios de inclusão e exclusão obrigatórios, com o seguinte processo de seleção:

- **Etapa 1**: A partir da leitura do título e resumo dos textos, os critérios serão avaliados com as seguintes regras:
  - A negativa adesão a um critério de inclusão ou a concordância com o critério de exclusão exclui imediatamente o estudo.
  - Um estudo só será selecionado para a próxima etapa se verificado sua adesão a todos os critérios de inclusão e não adesão a todos os critérios de exclusão estabelecidos.
- **Etapa 2**: A partir do resultante da etapa anterior, os critérios serão novamente aplicados após a leitura completa dos estudos, seguindo as mesmas regras.

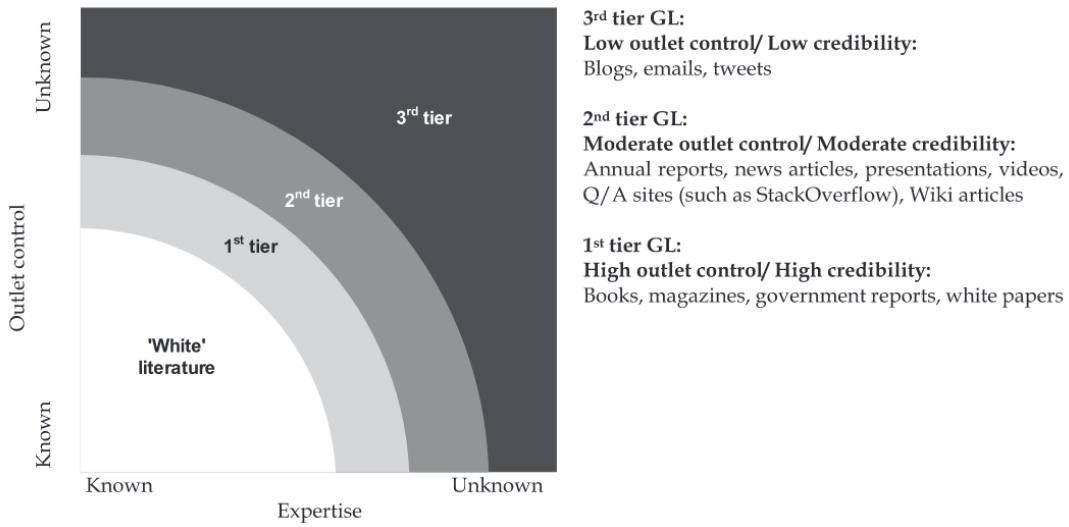
Na Tabela 4 encontram-se os critérios, definidos pela identificação (prefixo IN para Critério de Inclusão e EX para Critério de Exclusão, seguido do número). O critério em si e a categoria da evidência em seleção foram inspirados a partir da exposição de exemplos de critérios para revisões sistemáticas qualitativas (KITCHENHAM; BUDGEN; BRERETON, 2016), dos critérios usados no mapeamento sistemático sobre *design thinking* para atividades de desenvolvimento de software (PARIZI *et al.*, 2022), e nos critérios sugeridos para LC por Garousi, Felderer e Mäntylä (2019).

O critério EX7 usa uma categorização de evidência definida por Garousi, Felderer e Mäntylä (2019) com base em duas dimensões, especialização (*expertise*) e controle de saída ou tomada (*outlet control*), com mesma definição de limites, conhecido (*known*) e desconhecido (*unknown*), conforme é possível ver na Figura 3. Especialização é definida em "extensão em que a autoridade e o conhecimento do produtor do conteúdo podem ser

Tabela 4 – Critérios de Seleção de Estudo

ID	Critério	Categoria
IN1	Responder diretamente a uma ou mais questões de pesquisa	Todos
IN2	Relação com o contexto de segurança de software em startups	Todos
EX1	Ter sido escrito antes de 2014	Todos
EX2	Não escrito em inglês ou português	Todos
EX3	Não ser revisado por pares	LF
EX4	Não possuir relação com o contexto de segurança de software em startups	Todos
EX5	Não disponível para download ou acesso	Todos
EX6	Estudos duplicados	Todos
EX7	É classificado como nível 3 (baixa credibilidade) na Figura 3, com exceção a conteúdo de blogs	Todos

Figura 3 – "Tons" de LC. Fonte: Garousi, Felderer e Mäntylä (2019)



determinados" e controle de saída na determinação da conformidade em que o conteúdo é produzido, moderado ou editado de acordo com critérios transparentes de criação de conteúdo (GAROUSI; FELDERER; MäNTYLä, 2019). A LF é classificada como *White Literature*.

3.2.5 Avaliação de Qualidade

Segundo Kitchenham, Budgen e Brereton (2016), uma avaliação de qualidade "consiste em determinar até que ponto os resultados de um estudo empírico são válidos e isentos de preconceitos". A inclusão de estudos com baixa qualidade pode afetar a síntese de resultados, assim como aumentar o esforço durante a fase de extração de dados (KITCHENHAM; BUDGEN; BRERETON, 2016).

A definição da qualidade de um estudo é uma tarefa desafiadora e não possui uma definição acordada, mas a maioria das diretrizes e critérios estabelecidos a relacionam à busca de estudos em que design e execução "minimizam o viés e maximizam a validade" (KITCHENHAM; BUDGEN; BRERETON, 2016). Com base na definição exposta por Kitchenham, Budgen e Brereton (2016), a Tabela 5 apresenta conceitos importantes para a qualidade.

Tabela 5 – Conceitos de Qualidade adaptado de Kitchenham, Budgen e Brereton (2016)

Termo	**Sinônimos**	**Definição**
Viés	Erro sistemático	"...tendência para produzir resultados que se afastam sistematicamente dos resultados "verdadeiros". "Resultados imparciais são válidos internamente."
Validade interna	Validade	"A medida em que o desenho e a condução de um estudo são susceptíveis de evitar erros sistemáticos. A validade interna é um pré-requisito para a validade externa."
Validade externa	Generalização, aplicabilidade	"A medida em que os efeitos observados num estudo são aplicáveis fora do estudo."

Para avaliar os estudos, é necessária a definição de quais serão os critérios de qualidade julgados e como serão performados os procedimentos da avaliação. Ao definir critérios de qualidade, é necessário levar em conta quais serão os estudos avaliados, pois diferentes tipos de estudos demandam diferentes critérios relacionados à natureza e método de pesquisa (KITCHENHAM; BUDGEN; BRERETON, 2016).

Dybå e Dingsøyr (2008) desenvolveram critérios de qualidade baseados em princípios de boas práticas de pesquisa empírica de engenharia de software, compilados em um *checklist* que é largamente utilizado como base em avaliações de qualidade (KITCHENHAM; BUDGEN; BRERETON, 2016). Na Tabela 6 são apresentadas as quatro áreas de pesquisa empírica abrangidas pelos critérios definidos por Dybå e Dingsøyr (2008) e o relacionamento com os conceitos de qualidade apresentados na Tabela 5.

Para estudos terciários é necessário considerar critérios específicos para esse tipo de pesquisa, por isso, baseando-se na recomendação de Kitchenham, Budgen e Brereton

Tabela 6 – Definição e relação entre áreas empíricas de pesquisa, por Dybå e Dingsøyr (2008) e conceitos de qualidade por Kitchenham, Budgen e Brereton (2016)

Área de pesquisa empírica relacionada	**Relacionado a**	**Definição**
Relato (*Reporting*)	Validade interna	critérios relacionados a "qualidade do relato de um estudo empírico"
Rigor (*Rigour*)	Viés	critérios que "abordam detalhes do *design* da pesquisa"
Credibilidade (*Credibility*)	Validade interna	critérios focados em verificar se as "descobertas do estudo são válidas e significativas"
Relevância (*Relevance*)	Validade externa	critério que "diz respeito à relevância do estudo para a prática"

(2016), temos o critério DARE (*Database of Abstracts of Reviews of Effects*) que é formado por cinco perguntas, usado em domínio médico e aplicado em estudos terciários.

A LC também necessita de critérios próprios para avaliação, devido a processos de revisão e publicação menos controlados e mais diversos, tornando mais difícil sua avaliação.

A partir dos critérios apresentados por Dybå e Dingsøyr (2008), o uso do DARE (2002), da sugestão de critérios para LC por Garousi, Felderer e Mäntylä (2019) e da aderência quanto ao escopo dessa revisão, foram definidos os critérios de avaliação de qualidade que serão usados neste trabalho, agrupados no *checklist* presente na Tabela 7 e categorizados conforme tipo de estudo e área de pesquisa empírica.

O processo de avaliação da qualidade será conduzido a partir da aplicação do *checklist* em cada estudo selecionado (devido à categorização, será necessário extrair o tipo de estudo antes da aplicação), gerando um *score* que será utilizado para filtragem dos estudos e em fases posteriores da revisão, por exemplo, na síntese de resultados.

O *score* será calculado através da soma aritmética dos pontos obtidos em cada critério do *checklist*. Para isso, serão usadas duas escalas de pontos, definidas por:

1. Se aderente ao critério: 1 ponto, se parcialmente: 0,5 e se não: 0.
2. Se aderente ao critério: 1 ponto, se não: 0.

Em especial, o CQ1 servirá de critério excludente para a LF e não terá escala de pontuação, ou seja, os estudos não aderentes ao CQ1 serão descartados. O *score* máximo será de 9 para LF e 8 para LC, e os que possuírem *score* menor que 5 serão rejeitados.

3.2.6 Extração dos Dados

Para RML e RSL qualitativas a fase de extração de dados é comumente vinculada à fase de avaliação da qualidade. Nesse tipo de revisão busca-se a criação de uma base de dados de evidências, com a ferramenta de armazenamento e os extratores que serão usados claramente definidos (KITCHENHAM; BUDGEN; BRERETON, 2016).

Nesta revisão será usado um formulário para manter a consistência entre as fontes textuais e extratores, e guiar a extração dos dados textuais. Não será utilizada nenhuma ferramenta para extrair os dados, e estes serão armazenados em planilhas. Decorrente das recomendações por Kitchenham, Budgen e Brereton (2016), o formulário definido abrange as seguintes informações:

1. Item-alvo: definição do dado textual a ser buscado;
2. Valores: designação do dado textual;
3. QP: relação que indica a qual ou quais questões de pesquisa são respondidas pelo dado.

Os itens-alvo escolhidos buscam informações da publicação, data da extração e de como ela responde às questões de pesquisa da revisão. Na Tabela 8 encontra-se o formulário criado.

3.2.7 Síntese

A síntese compreende uma atividade de resumo, integração, combinação e comparação dos dados obtidos na prática anterior (KITCHENHAM; BUDGEN; BRERETON, 2016). Para esta atividade, uma síntese qualitativa será empregada, devido a expectativa das fontes textuais a serem sintetizados utilizarem metodologias de pesquisa qualitativa, ou de possuírem diferentes metodologias experimentais (KITCHENHAM; BUDGEN; BRERETON, 2016).

O método escolhido será o de análise temática, definido por Kitchenham, Budgen e Brereton (2016), como um método que "envolve a identificação e codificação dos temas principais ou recorrentes nos estudos primários e o resumo dos resultados sob esses títulos temáticos.", e identificado como o segundo método qualitativo mais usado por pesquisadores de engenharia de software (KITCHENHAM; BUDGEN; BRERETON, 2016).

A partir da adaptação das cinco etapas definidas por Cruzes e Dyba (2011), para uma síntese temática, as seguintes etapas foram estipuladas, com início após a atividade precedente de extração dos dados:

1. Organização e Examinação: Processo para rotulação dos dados de fonte, evoluindo para a organização e agrupamento de dados similares (que compartilham alguma característica) em categorias.

2. Análise para redução de sobreposições e Definição de temas: Processo em que diferentes rótulos são combinados para formação de temas abrangentes, atividade realizada em ciclos com o objetivo final de exaurir todos os temas possíveis dos dados.

3. Criação de Modelos entre temas de ordem superior: Exploração dos temas para criação de um modelo com relacionamento entre temas de ordem superior. Com objetivo de "retornar às questões originais da pesquisa e aos interesses teóricos que as sustentam, e abordá-las com argumentos fundamentados nos temas que surgiram na exploração dos textos".

Tabela 7 – *Checklist* da avaliação da qualidade

ID	Área de pesquisa empírica relacionada	Escala de pontos	Tipo de estudo	Critério
CQ1	Relato	-	LF	"O artigo é baseado em pesquisa (ou é apenas um relatório de "lições aprendidas" baseado na opinião de especialistas)?" (DYBå; DINGSøYR, 2008)
CQ2	Relato	Sim(1) Parcial(0,5) Não(0)	LF	"Existe uma declaração clara dos objetivos da pesquisa?" (DYBå; DINGSøYR, 2008)
CQ3	Relato	Sim(1) Parcial(0,5) Não(0)	LF	"Existe uma descrição adequada do contexto em que a pesquisa foi realizada?" (DYBå; DINGSøYR, 2008)
CQ4	Rigor	Sim(1) Parcial(0,5) Não(0)	LF (Primário)	"O desenho da investigação foi apropriado para abordar os objetivos da investigação?" (DYBå; DINGSøYR, 2008)
CQ5	Rigor	Sim(1) Parcial(0,5) Não(0)	LF (Primário)	"A estratégia de recrutamento foi adequada aos objetivos da investigação?" (DYBå; DINGSøYR, 2008)
CQ6	Rigor	Sim(1) Não(0)	LF (Primário)	"Os dados foram recolhidos de forma a abordar o problema da investigação?" (DYBå; DINGSøYR, 2008)
CQ7	Rigor	Sim(1) Parcial(0,5) Não(0)	LF (Primário)	"A análise dos dados foi suficientemente rigorosa?" (DYBå; DINGSøYR, 2008)
CQ8	Credibilidade	Sim(1) Parcial(0,5) Não(0)	LF (Primário)	"A relação entre pesquisador e participantes foi considerada adequadamente?" (DYBå; DINGSøYR, 2008)
CQ9	Credibilidade	Sim(1) Parcial(0,5) Não(0)	Todos	"Existe uma declaração clara das conclusões?" (DYBå; DINGSøYR, 2008)
CQ10	Relevância	Sim(1) Não(0)	Todos	"O estudo tem valor para pesquisa ou prática?" (DYBå; DINGSøYR, 2008)
CQ11	Rigor	Sim(1) Parcial(0,5) Não(0)	LF (Secundário e Terciário)	"Os critérios de inclusão e exclusão da revisão estão descritos e apropriados?" (DARE, 2002)
CQ12	Rigor	Sim(1) Não(0)	LF (Secundário e Terciário)	"É provável que a pesquisa bibliográfica tenha coberto todos os estudos relevantes?" (DARE, 2002)
CQ13	Credibilidade	Sim(1) Não(0)	LF (Secundário e Terciário)	"Os revisores avaliaram a qualidade/validade dos estudos incluídos?" (DARE, 2002)
CQ14	Rigor	Sim(1) Parcial(0,5) Não(0)	LF (Secundário e Terciário)	"Os dados/estudos básicos foram descritos adequadamente?" (DARE, 2002)
CQ15	Rigor	Sim(1) Não(0)	LF (Secundário e Terciário)	"Os estudos incluídos foram sintetizados?" (DARE, 2002)
CQ16	Credibilidade	Sim(1) Não(0)	LC	"A organização de publicação ou a organização associada ao autor é respeitável? O autor publicou outros trabalhos ou tem experiência na área?" (GAROUSI; FELDERER; MäNTYLä, 2019)
CQ17	Rigor	Sim(1) Parcial(0,5) Não(0)	LC	"A fonte tem um objetivo e uma metodologia claramente declarada? Tem referências contemporâneas e confiáveis? Há algum limite claramente declarado? O trabalho cobre uma questão específica? O trabalho se refere a uma população ou caso específico?" (GAROUSI; FELDERER; MäNTYLä, 2019)
CQ18	Relato	Sim(1) Parcial(0,5) Não(0)	LC	"O trabalho parece equilibrado na apresentação? A declaração nas fontes é o mais objetiva possível? Há interesse pessoal? As conclusões são apoiadas pelos dados?" (GAROUSI; FELDERER; MäNTYLä, 2019)
CQ19	Relato	Sim(1) Parcial(0,5) Não(0)	LC	Pertence ao nível 1 conforme Figura 3
CQ20	Relevância	Sim(1) Parcial(0,5) Não(0)	LC	"Isso enriquece ou acrescenta algo único à pesquisa? Fortalece ou refuta uma posição atual?" (GAROUSI; FELDERER; MäNTYLä, 2019)
CQ21	Rigor	Sim(1) Parcial(0,5) Não(0)	LC	"As principais fontes formais ou LC relacionadas foram vinculadas/discutidas?" (GAROUSI; FELDERER; MäNTYLä, 2019)

Tabela 8 – Formulário de extração de dados

Item-alvo	**Valore(s)**	**Comentário**
Data da extração	-	Não se aplica
Título da publicação	-	Não se aplica
Data da fonte	-	Não se aplica
Autores	-	Não se aplica
Base de dados	ACM Digital Library IEEE Digital Library Google Manual por conferência Scopus	Não se aplica
Tipo da fonte	LF (Primário, Secundário ou Terciário) LC	Não se aplica
Tipo de pesquisa	Proposta de solução, pesquisa de validação (estudo empírico fraco), pesquisa de avaliação (estudo empírico forte), estudos de experiência, estudos filosóficos, estudos de opinião, outros	Não se aplica
Práticas de Segurança de Software	Nome da prática	QP1
Descrição das Práticas de Segurança de Software	Descrição do que consiste a prática de segurança	QP1
Contexto de aplicação das Práticas de Segurança de Software	Onde e como a prática de segurança é aplicada	QP1 e QP2
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Experiência, opinião e evidências esperadas ou encontradas após aplicação da prática de segurança	QP2
Dificuldades encontradas	Dificuldades identificados para aplicação das práticas de segurança	QP3

4 Resultados

Com o protocolo da revisão definido na Fase 1, Seção 3.2, é chegada a hora de partir para a condução, a Fase 2 de uma RML. Neste Capítulo, serão apresentados e comentados os resultados obtidos durante a condução da revisão, divididos nas seguintes seções:

1. Processo de Busca, Seção 4.1;
2. Seleção dos estudos, Seção 4.2;
3. Avaliação da qualidade, Seção 4.3;
4. Extração dos dados, Seção 4.4;
5. Síntese, Seção 4.5.

4.1 Processo de Busca

Após aplicada a *string* de busca nas bases IEEE, ACM e Scopus, conforme Tabela 3, as evidências resultantes foram coletadas e inseridas na ferramenta online Parsifal, concebida "[...] para auxiliar os investigadores a realizar revisões sistemáticas de literatura no contexto da Engenharia de Software" (PARSIFAL, 2025). As evidências coletadas a partir da busca manual por conferência, ao todo duas, também foram adicionadas ao Parsifal.

Com a busca automatizada na base Google foram coletadas e inseridas 73 evidências no Parsifal, resultantes da aplicação da string de busca e limitadas conforme definição de parada. Os 70 primeiros resultados atingiram o limite de esforço, e, para o restante, a análise de mais 90 evidências levou à saturação teórica.

O somatório de evidências coletadas é igual a 259. Na Figura 4 é possível observar um gráfico de setores com a representação da fração de evidências por base pelo total.

Na Figura 5 é apresentado um gráfico de linha com a distribuição das evidências por ano. Nota-se uma tendência de aumento nos últimos anos, devido ao fomento da discussão sobre segurança de software e à inclusão de LC na seleção de estudos.

4.2 Seleção dos estudos

Com base nos critérios definidos na Tabela 4, sua aplicação resultou em um montante de 33 evidências aceitas e selecionadas para avaliação de qualidade. Em relação aos

Figura 4 – Evidências coletadas por base de dados. Fonte: Autor

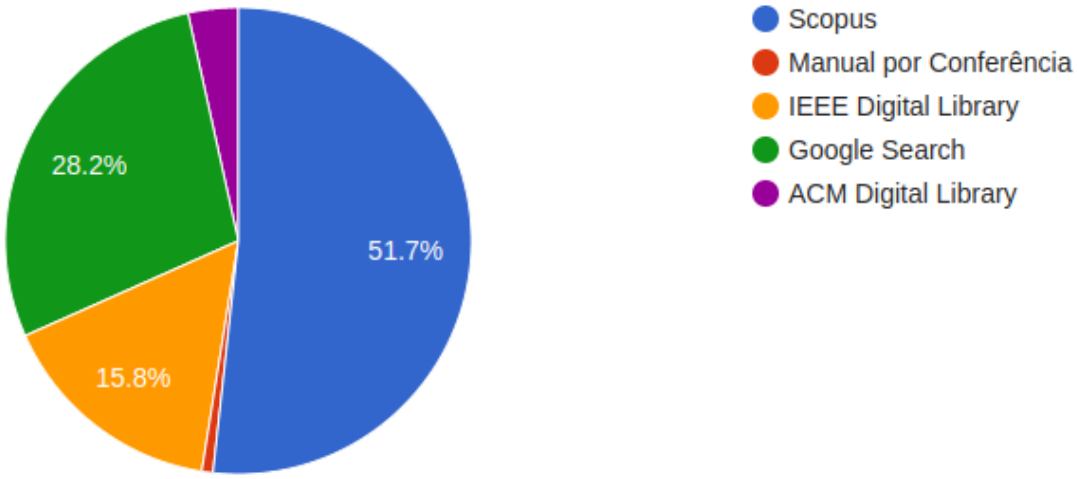


Figura 5 – Evidências por ano. Fonte: Autor

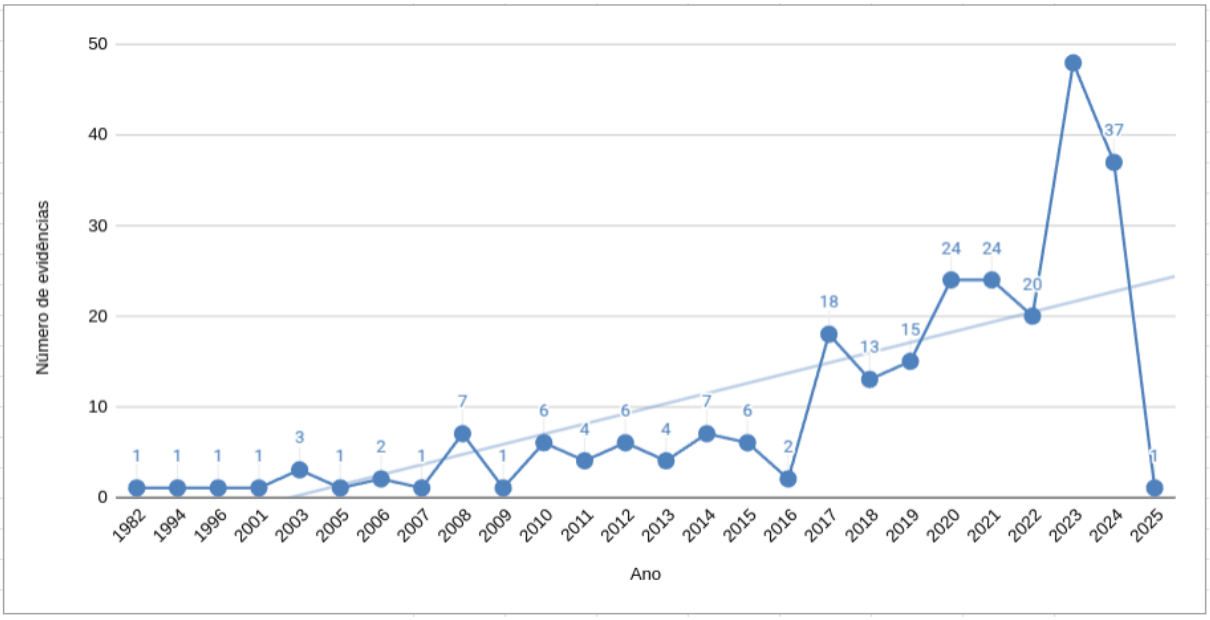
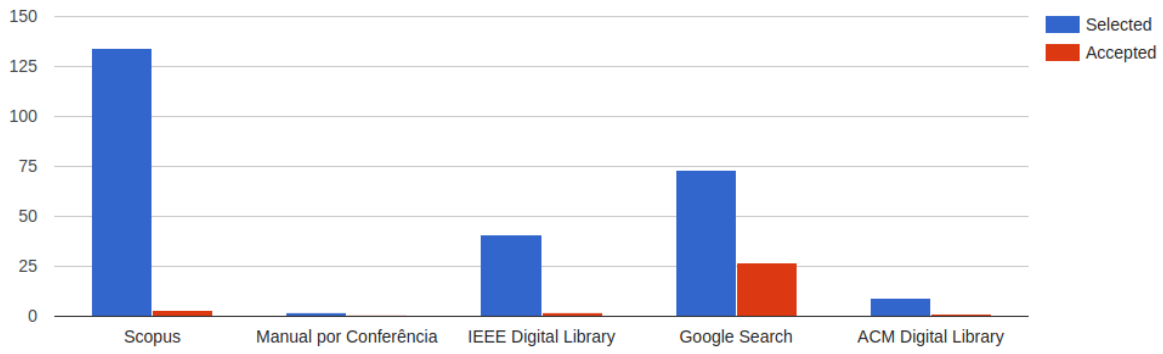


Figura 6 – Evidências selecionadas e aceitas por base de dados. Fonte: Autor



eliminados, 120 foram eliminados pelo critério de exclusão EX4, 35 pelo EX1, 29 pelo EX5, 27 pelo EX6 e 15 pelo EX7. Na Figura 6 é apresentado um gráfico de barras verticais, que representa a quantidade de evidências selecionadas e aceitas por base.

4.3 Avaliação da Qualidade

O processo de avaliação de qualidade aplicou para cada evidência selecionada um *checklist*, conforme Tabela 7. Após a avaliação, 19 evidências obtiveram *score* maior que 5,0, sendo três com *score* máximo, e o restante, 13, foi rejeitado com *score* menor que 5,0. Na Tabela 9 estão listadas as evidências avaliadas, com respectivo identificador (ID), evidência, base de dados de origem e score atribuído.

Na Tabela 10 está consolidado o somatório de evidências aceitas à base de origem.

4.4 Extração dos dados

A extração dos dados foi realizada para cada uma das 19 evidências, de acordo com o formulário presente na Tabela 8. Para cada evidência, foi criada uma tabela com os dados obtidos, apresentada no Apêndice B.

Com o processo de extração dos dados, foi percebido que as evidências caracterizadas como LC, em sua maioria, apresentam definição, contextualização e exemplificação rasas de práticas de segurança. Alguns autores as apresentam como “bala de prata” para a promoção de segurança de software dentro da empresa e para o desenvolvimento de um produto seguro, sem uma apresentação clara diretamente relacionada entre contexto e dificuldade da aplicação da prática em si. Ademais, algumas evidências possuem caráter enviesado para promoção de serviço ou software relacionado a alguma prática de segurança. LC representa 68% das evidências.

Tabela 9 – Resultado da Avaliação de Qualidade

ID	Evidência	Base de dados	*Score*
E1	"We are a startup to the core": A qualitative interview study on the security and privacy development practices in Turkish software startups. (KEKüLLüOğLU; ACAR, 2023)	Scopus	9
E2	11 Cybersecurity Tips Every Startup Should Follow (SUNDAR, 2017)	Google	6,5
E3	5 Steps to Put Your Startup On the Path Towards Better Security e (JEBARA, 2019)	Google	3,0
E4	7 Ways to Make Sure Your Digital Startup More Secure (JAIN, 2024)	Google	2,5
E5	A Developer's Guide to Startup Securit: 15 Ways to Secure Your Startup (WORKOS, 2021)	Google	6,5
E6	A Security Checklist for Your Startup (KELLER, 2021)	Google	5
E7	Accomplishing More With Less: The Practice of Cybersecure Health Technology Design Among Danish Startups (LI *et al.*, 2023)	Scopus	9
E8	Application security made simple: Inside Belgian startup Aikido Security's "no BS" security platform and European focus – BeBeez International (CANALS, 2024)	Google	2
E9	Basic Security Practices and Features for A Startup Software Product (TEAM, 2020)	Google	5,5
E10	Build an information security program that scales with your startup (GAINES, 2024)	Google	6
E11	Building Cybersecurity Strong Defenses for Startups (NOVIKOV, 2024)	Google	6,5
E12	Creating a security program for a startup (BUTTERSCOTCHAPART500, 2021)	Google	2,5
E13	Essentials of Secure Code Writing: Key Strategies for Startup Success (MOQOD, 2024)	Google	5,5
E14	Exploring the Quantum Computing Benefits on Startup Innovation and Security (SCHLOPSNA, 2024)	Google	6,5
E15	How to Build a Robust Cybersecurity Strategy for Your Startup (NICK, 2023)	Google	5,5
E16	Importance of Application Security and Customer Data Protection to a Startup (NEWS, 2021)	Google	2,0
E17	Mental models for cybersecurity startup founders and why Microsoft cannot easily force nimble early-stage companies out of business (HALELIUK, 2023)	Google	4,0
E18	Penetration Testing: A Cost-Benefit Analysis of Best Practices Implementation for Software Startups (GAAFAR; FOUAD; SADEK, 2024)	IEEE	8,5
E19	Security at Early Stage Startup (SHARMA, 2023)	Google	1
E20	Security at Startup (KRAVCENKO, 2023)	Google	4,5
E21	Security Evaluation by Arrogance: Saving Time and Money (ALMOHRI; ALMOHRI, 2017)	ACM	7,5
E22	Seven Cybersecurity Tips To Strengthen Your Startup's Security Posture (RENDE, 2024)	Google	1,5
E23	Startup DevSecOps Security (BANSAL, 2023)	Google	5,5
E24	Startup Security: A Framework From Series B to F Funding (WHITAKER, 2023)	Google	6
E25	Startup Software Development Guide (How We Build Successful Apps) (KOVALENKO, 2024)	Google	2,5
E26	Strategies for a Startup Software-as-a-Service Organizations with Minimal Budget to Achieve Security and Compliance Goals (LANKA; VAROL; SHASHIDHAR, 2023)	IEEE	9
E27	Strategy Indicators for Secure Software Development Lifecycle in Software Startups Based on Information Security Governance (FERDIANSYAH; ISNANTO; SUSENO, 2023)	Scopus	8,5
E28	The essential cybersecurity tools every startup needs to stay secure — TFN (NEWS, 2024)	Google	4,5
E29	The Startup 7: Cybersecurity Fundamentals for Early Stage Startups (SAGARD, 2020)	Google	6
E30	Top 10 Tips to Secure your Startup (CYBERTALENTS, 2020)	Google	1,5
E31	What security needs and what type of pentest for a startup? (VAADATA, 2021)	Google	6
E32	What Type of Software Testing Should a Startup Focus on First? (K, 2024)	Google	4,5
E33	What you need to know about startup security (FASTERCAPITAL, 2024)	Google	3

4.5 Síntese

Conforme definido na Seção 3.2.7, o processo de Síntese será composto por uma análise temática qualitativa.

Tabela 10 – Relação entre base de dados e quantitativo com score maior ou igual a 5.0

Base de Dados	**Número de evidências**
ACM Digital Library	1
IEEE Digital Library	2
Google	13
Manual por conferência	0
Scopus	3

4.5.1 Organização e Examinação

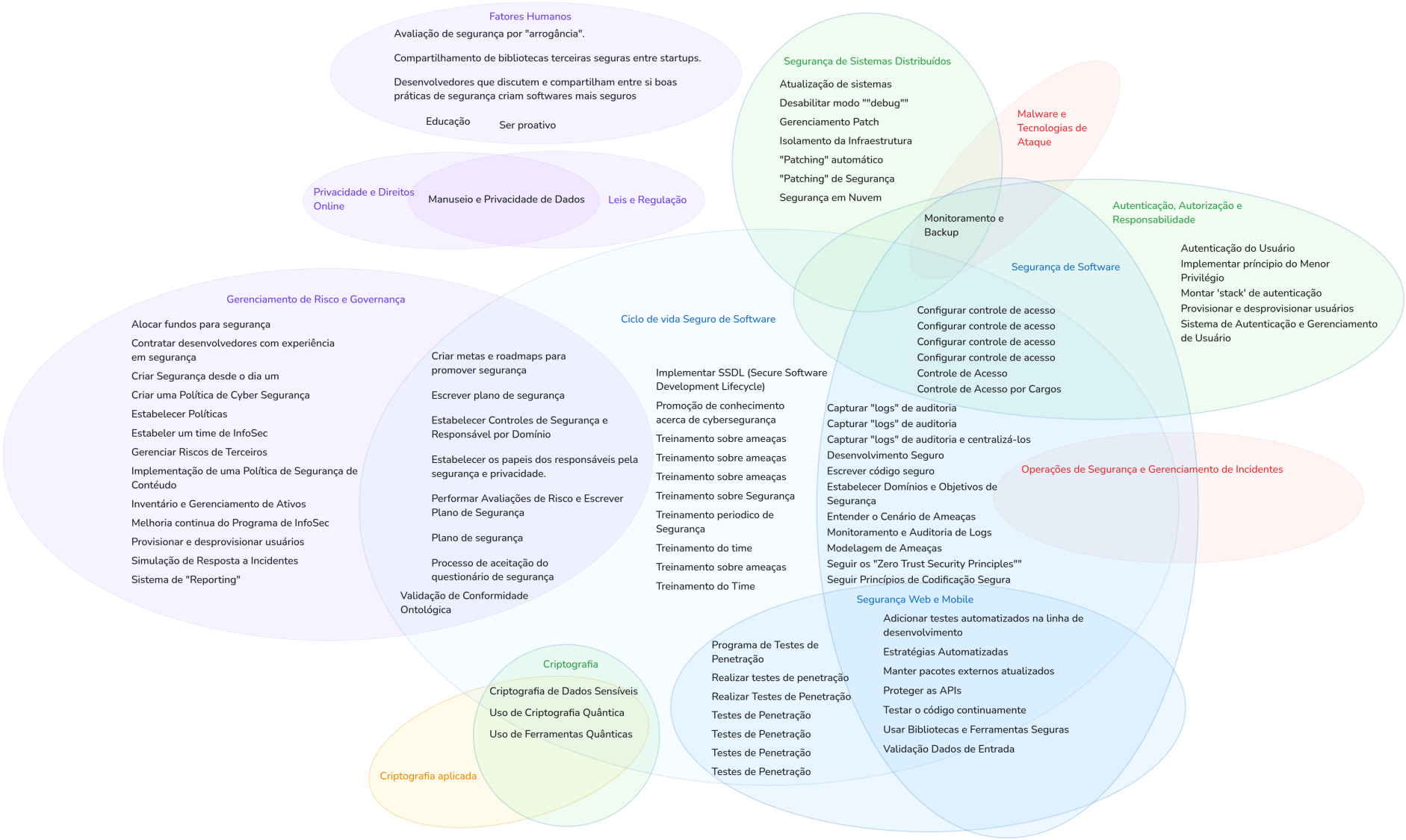
O processo de Organização começou com a leitura completa das evidências. A extração dos dados voltada para responder às Questões de Pesquisa deu início à Examinação.

Os processos de Organização e Examinação foram contemplados pela condução da revisão, desde a seleção das evidências até a extração dos dados, resultando no agrupamento de dados que possibilitam a resposta às QP.

4.5.2 Análise para redução de sobreposições e Definição de temas

A partir das práticas de segurança extraídas, foi feito um diagrama de Venn que as agrupa por temas que representam áreas de conhecimento de baixo nível do escopo CyBoK (conforme Figura 1). O diagrama está presente na Figura 7. A Figura 7 revela que grande parte das práticas estão relacionadas ao Ciclo de Vida Seguro de Software, seguido da Segurança de Software, que apresentam uma grande correlação entre práticas.

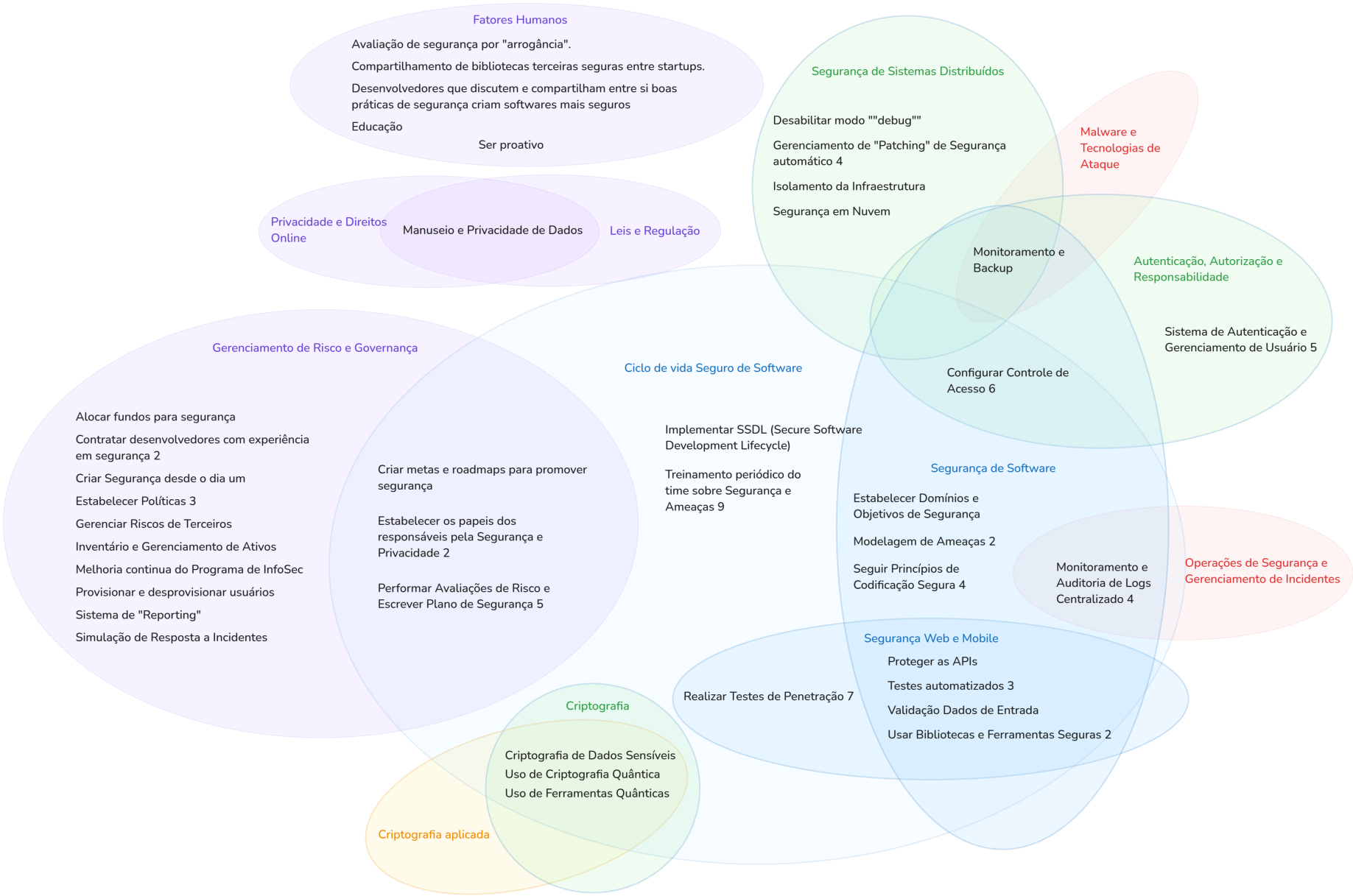
Figura 7 – Práticas agrupadas por áreas de conhecimento de baixo nível CyBoK. Fonte: Autor



Conforme análise da Figura 7, nota-se uma repetição de termos que representam as mesmas práticas de segurança. Devido a essa repetição, um processo de análise foi realizado para obter a ocorrência das práticas, para que um novo diagrama enxuto pudesse ser criado (Figura 8) com a representação da prática e seu número de ocorrências. As práticas que não possuem número explícito possuem ocorrência igual a uma. A partir da Figura 8 percebe-se quais práticas tiveram maior frequência, sendo elas, em ordem decrescente:

- Nove ocorrências: Treinamento periódico do time sobre Segurança e Ameaças.
- Sete ocorrências: Realizar Testes de Penetração.
- Seis ocorrências: Configurar Controle de Acesso.
- Cinco ocorrências: Performar Avaliações de Risco e Escrever Plano de Segurança.
- Quatro ocorrências: Monitoramento e Auditoria de Logs Centralizado; Gerenciamento de ”Patching“de Segurança Automático; Seguir Princípios de Codificação Segura;
- Três ocorrências: Testes automatizados; Estabelecer Políticas;
- Duas ocorrências: Contratar desenvolvedores com experiência em segurança; Estabelecer os papéis dos responsáveis pela Segurança e Privacidade; Modelagem de Ameaças; Usar Bibliotecas e Ferramentas Seguras;

Figura 8 – Práticas combinadas e agrupadas por áreas de conhecimento de baixo nível CyBOK. Fonte: Autor



Com base nas informações extraídas a respeito das dificuldades encontradas para a aplicação de práticas de segurança, foi feito um agrupamento por temas, apresentado na Figura 9. A criação dos temas levou em consideração problemas semelhantes relatados pelos dados, como: interesse por segurança de software; recursos e retorno de investimento; conhecimento técnico e literatura; desenvolvimento das práticas de segurança; relação entre a empresa e a segurança de software.

Observa-se pela Figura 9 que as dificuldades como: falta ou escassez de conhecimento técnico; desinteresse por segurança pelo desenvolvedores; falta de priorização de requisitos de segurança; aumento no tempo de desenvolvimento; encontrar código de terceiros seguro; pensamento tardio em segurança no design de sistemas; alocação de recursos e custo de implementação.

Além das práticas e dificuldades, um dado importante para responder às QPs é o contexto de aplicação das práticas de segurança. Os dados extraídos apresentam pouca variação e alta repetição, não permitindo o agrupamento em temas abrangentes e divergentes. Na Figura 10 os dados referentes ao contexto são reunidos.

4.5.3 Criação de Modelo entre temas de ordem superior

Na Figura 11 é apresentado um diagrama onde os temas da Figura 8 foram combinados em temas de nível superior, com base no CyBoK. Com a combinação em temas de nível superior é possível perceber, a partir da Figura 11, que grande parte das práticas estão classificadas nas áreas de Segurança de Software e Plataforma, e Aspectos Humanos, Organizacionais e Regulatórios, representando 85% de todas as práticas.

4.6 Análise

O processo anterior de síntese gerou insumos que permitem analisar os resultados obtidos e se eles respondem às QP. Primeiramente, é necessário relembrar as QP definidas para a RML:

- QP1 - Quais as práticas de segurança de software tem sido utilizada por *startups*?
- QP2 - Quais práticas de segurança de software são mais adaptáveis para *startups* em estágio inicial?
- QP3 - Quais as principais dificuldades enfrentadas pelas *startups* para aplicar as práticas de segurança?

As práticas obtidas, conforme Figura 11, apresentam convenções já conhecidas da Engenharia de Software, com exceção da prática de Avaliação de Segurança por “arro-

Figura 9 – Dificuldades agrupadas por tema. Fonte: Autor

Interesse por Segurança de Software

* "houve falta de interesse em segurança na maioria dos outros engenheiros de software." PG 13

* Desenvolvedores não interessados nas práticas de DevSecOps

Conhecimento Técnico e Literatura

* "Além disso, para equipes de startups, pesquisar, aprender e aplicar corretamente um processo teórico como o Ciclo de Vida de Desenvolvimento de Segurança da Microsoft [5] pode não ser viável". PG 12

* Escassez de conhecimentos técnicos

* "Infelizmente, a literatura de pesquisa tem uma visão limitada de metodologias e diretrizes para avaliação de segurança de software desenvolvido em ambientes de startups". PG 12

* Conhecimento técnico

Relação Entre a Empresa e Segurança de Software

* Priorização de requisitos de segurança quase que somente por força de regulamentações externas a empresa

* Na abordagem de governança do SSDL, os cargos de CEO e CTO são de muita responsabilidade e críticos no alcance da implementação do processo de desenvolvimento seguro de software, o que indica que se um deles falhar a segurança será muito comprometida.

* Implementar o programa pode ser desafiador, mas é necessário apenas começar."

* "Entender quais aspectos da startup precisam estar seguros"

* "Mercado não conhece os benefícios da computação quântica"

* "Entender o porquê uma startup precisa de um programa de InfoSec.

Desenvolvimento

*Integração entre Sistemas Complexa

* Aumento do tempo de desenvolvimento

* "A intensidade do desenvolvimento, as rápidas mudanças nos requisitos e a pressão sobre a equipe para gastar o mínimo de tempo no desenvolvimento de um componente, estavam entre os motivos que atrasaram um teste completo de segurança tanto das aplicações quanto dos serviços backend." PG 13

* A utilização do modelo não garante a conformidade com padrões ISO ou IEC

* Encontrar bibliotecas (códigos de terceiros) seguros

* Segurança não é algo pensado no design dos sistemas, só quando o sistema amadurece

* "A escolha da arquitetura que precisa ser implementada para um controle específico depende da facilidade de implementação, da complexidade do código e dos recursos da plataforma do provedor de serviços em nuvem."

* "Não é possivel automatizar todos os controles de segurança dentro das tecnologias usadas na aplicação, requerindo processo manual de implementação."

Recursos e Retorno de Investimento

* Alocação de recursos

* Adaptar o orçamento do cliente ao custo dos testes.

* Necessidade de Alto Investimento

* Custo inicial considerável para contratação de especialistas em testes de penetração e da implementação de boas práticas de segurança.

* Alto custo

* Orçamento limitado

* Acesso limitado a recursos

* Retorno incerto de investimentos

Figura 10 – Contexto de aplicação das práticas de segurança. Fonte: Autor

* "Uma startup de software que desenvolvia um sistema de pagamento mobile. 2012. Time de 4 engenheiros. Não sabiam como estava a qualidade do software lançado a cada release, tampouco se respeitavam as políticas de segurança dispostas aos clientes. Questionavam se seria possível uma ""avaliação de segurança com custo quase zero"". ""Nossa hipótese era que um engenheiro arrogante com uma atitude tecnicamente negativa em relação aos colegas pode ser uma vantagem em vez de um fardo"" PG 12

* "Startup com 50-150 funcionários. Nenhum/Um/Alguns especialistas em segurança. Clientes consideram o produto/serviço de alto risco dentro do modelo de ameaças. Alguns funcionários possuem acesso a todos os dados sensíveis"

* Startups de software, e pequenas e médias empresas dependentes de dispositivos e aplicativos móveis.

* Startups que possuem recursos escassos e pouco pessoal especializado em segurança.

* Startups que já realizam ou querem realizar testes de penetração

* Startup de software turcas

* Startups que desenvolvem soluções tecnológicas de saúde

* Startups em estágio inicial

gância" (E21), denotando que as evidências majoritariamente remontam à literatura no momento de introduzir ou recomendar práticas de segurança de software.

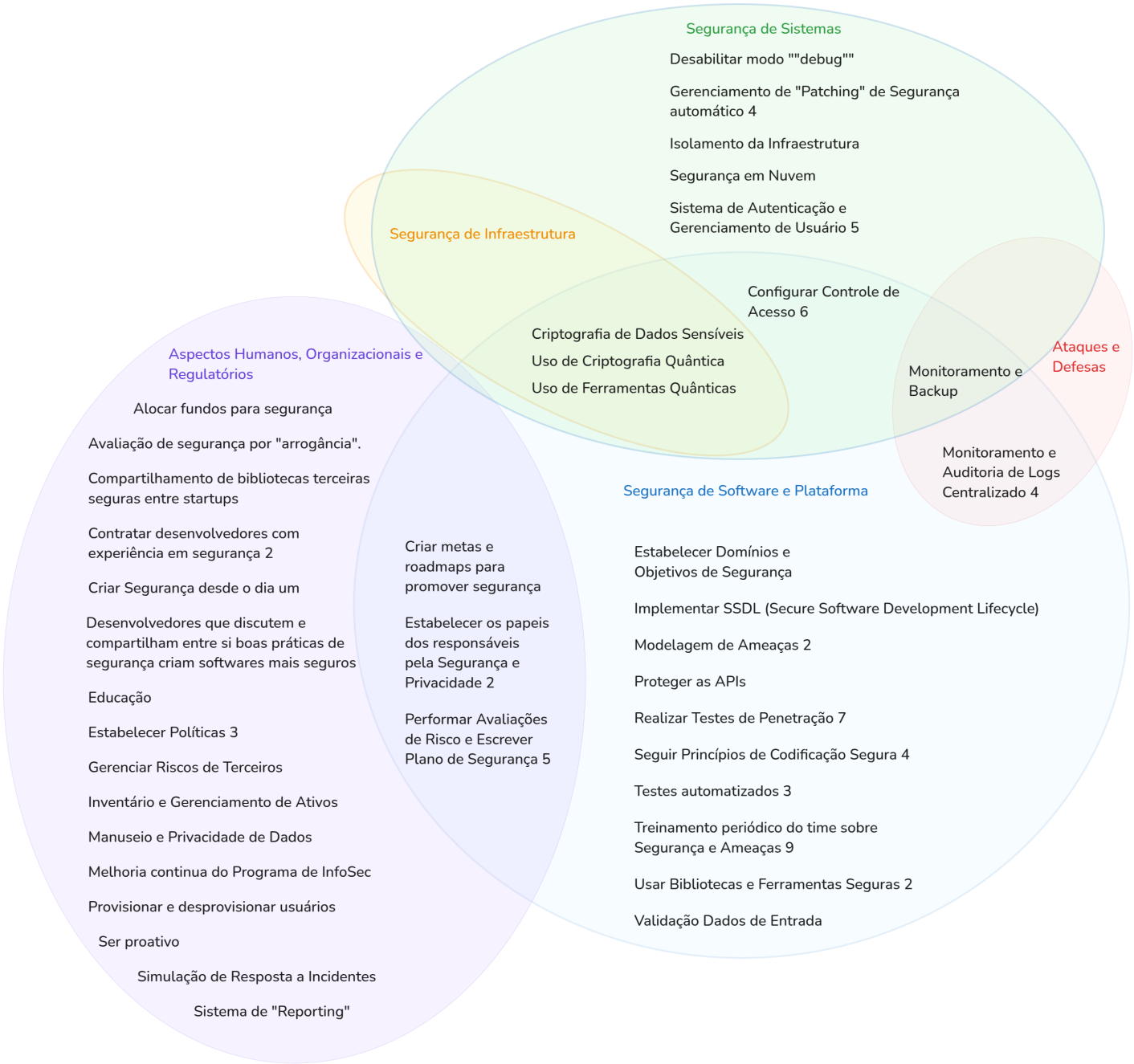
Cerca de 62% das práticas são encontradas nos ciclos de vida de software seguro SDL e SAFECode. Das práticas extraídas pertencentes à área de conhecimento Aspectos Humanos, Organizacionais e Regulatórios, uma área que não envolve diretamente o desenvolvimento de código, têm-se 32%.

A realização da RML permitiu responder a QP1, sendo que a seguir têm-se as seis práticas de segurança mais frequentes atribuídas a *startups*:

1. Treinamento periódico do time sobre Segurança e Ameaças (nove ocorrências): Envolve o treinamento periódico do time sobre segurança de software (E23), melhores práticas de segurança (E2), ameaças e como se defender (E5), vulnerabilidades (E24), criação de uma mentalidade de segurança no time (E15), ferramentas para promoção de conhecimento (E7), reconhecer ataques (E10).

2. Realizar Testes de Penetração (sete ocorrências): Realização de testes de penetração alinhados com o ciclo de entregas, escopo de análise e risco de exposição da empresa (E31). Importante para encontrar fraquezas no sistema antes de um possível ataque (E15).

Figura 11 – Práticas combinadas e agrupadas por áreas de conhecimento de alto nível CyBOK. Fonte: Autor



3. Configurar Controle de Acesso (seis ocorrências): Permitir acesso a dados e recursos de acordo com características cargo (E2, E5), com o uso de autenticação e autorização (E11).

4. Performar Avaliações de Risco e Escrever Plano de Segurança (cinco ocorrências): Plano de resposta a incidentes com construção de um passo a passo de resposta imediata (E5, E24); identificar ameaças e vulnerabilidades para cada ativo e desenvolver planos para mitigar, transferir ou aceitar os riscos; plano de ação caso seja atacado (E10).

5. Monitoramento e Auditoria de Logs Centralizado (quatro ocorrências): Armazenar e processar eventos estruturados para análise dos dados (E5); manter logs centralizados fora da infraestrutura principal, para que seja possível correlacionar eventos, buscar funcionalidades e investigar problemas (E11); monitorar sistemas em produção e revisar logs periodicamente (E13).

6. Seguir Princípios de Codificação Segura (quatro ocorrências): Seguir os seguintes príncipios: controle de acesso; validação de dados de entrada das aplicações e criptografia de dados sensíveis (E13); desenvolver código de maneira que esteja salvo de introdução de vulnerabilidades (E9); adicionar segurança como requisito na arquitetura de aplicações (E29).

Em relação a QP2 os resultados obtidos na Figura 10 demonstram que o contexto de aplicação das práticas aconteceu ou é indicado majoritariamente em *startups* de estágio inicial, sendo assim é possível relacionar que as práticas mais frequentes abordadas no parágrafo anterior são as mais adaptáveis para *startups* nesse contexto.

Quanto a QP3 os resultados achados, Figura 9, reforçam ainda mais o Problema elencado neste trabalho (Seção 1.2), mostrando que o desafio encontrado por *startups* percorre desde a alocação de recursos até o desenvolvimento das práticas de segurança. Dentre as dificuldades mais frequentemente relatadas têm-se a alocação de recurso para segurança e a deficiência de conhecimento técnico tanto de desenvolvedores como de papeis ligados a governança e liderança de empresas. Por falar em liderança, a evidência E27 relata uma sobrecarga centrada nos papéis do CTOs e CEOs para o sucesso de implementação do desenvolvimento seguro de software e da segurança do produto.

Sendo assim com a realização da RML foi possível responder as QP com sucesso.

5 Conclusão

Este trabalho teve como objetivo geral identificar práticas de segurança de software para *startups* em estágio inicial, sendo esse objetivo desmembrado em dois objetivos específicos: identificar práticas de segurança viáveis de implementação no contexto de *startups* em estágio inicial (OE1), e organizar as práticas de segurança por critérios similares usados na engenharia de software moderna (OE2). A construção do objetivo se deu pela problemática enfrentada por *startups* com poucos recursos financeiros e de mão de obra qualificada para encontrar práticas de segurança adaptáveis a sua realidade a fim de construir soluções seguras.

A execução de uma RML forneceu resultados que indicam práticas que se moldam ao estágio inicial de uma *startup* de software, contudo os desafios de implementação ainda persistem, exigindo um alto investimento inicial em segurança, profissionais qualificados e a instauração de um programa de segurança desde o dia um. Os fatores humanos como conhecimento técnico, proatividade e divulgação de conhecimento entre funcionários também foram frequentemente relatados como empecilhos.

A organização de práticas de segurança por critérios similares se deu pela distribuição dos resultados por área de conhecimento do CyBoK, em destaque as áreas de Segurança de Software e Plataforma, e Aspectos Humanos, Organizacionais e Regulatórios, como as que mais apresentaram quantitativo de práticas.

Este trabalho contribui com a pesquisa em Engenharia de Software e a indústria, reforçando a necessidade de pesquisa aplicada a práticas de segurança de software em *startups* e ao conhecimento de práticas ajustadas ao contexto de *startups*, respectivamente.

5.1 Limitações do Trabalho

A maior parte dos resultados obtidos não fornecem material suficiente para que dificuldade e contexto estejam diretamente vinculados a uma específica prática de segurança, o que culminou na remoção de um dos objetivos específicos de gerar um guia de boas práticas de segurança.

O material investigado classificado como LF obteve uma presença de apenas 31% (ao todo seis evidências), que demonstra a pouca produção de literatura acerca do tema, o que também afetou na qualidade geral das evidências analisadas. As práticas extraídas da LF apresentaram rica descrição de contexto, desafios e descrição em si das práticas, além de inovarem no promoção de novas abordagens de segurança em *startups*.

5.2 Trabalhos Futuros

Para futuros trabalhos, a fim de continuar a investigação na temática de boas práticas de segurança para *startups*, são elencados pontos que agregariam maior valor para este e outros estudos:

- Melhoria na Metodologia:
  - Seleção rigorosa de LC, devido ao alto viés de propaganda e promoção de serviço próprio.
  - Alterar *string* de busca que remova o duplo sentido da palavra *startup* (durante a seleção das evidências, muitas se referiam a *startup* como o processo de inicialização de um software).
- Criação de um guia de boas práticas de segurança para *startups* em estágio inicial:
  - A criação de um guia pode ajudar a indústria de *startups* a promover a segurança em seus processos e produtos.
  - Com o avanço em pesquisa nessa área a criação de um guia pode ser possível.

Referências

ALMOHRI, H. M. J.; ALMOHRI, S. A. Security evaluation by arrogance: Saving time and money. In: *Proceedings of the 1st International Workshop on Software Engineering for Startups*. [S.l.]: IEEE Press, 2017. (SoftStart '17), p. 12–16. ISBN 9781538628010. Citado na página 62.

BANSAL, A. Blog, *Startup DevSecOps Security*. 2023. Disponível em: <https://www.clouddefense.ai/startup-devsecops-security/>. Citado na página 62.

BERG, V. *et al.* Software startup engineering: A systematic mapping study. v. 144, p. 255–274, 2018. ISSN 01641212. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0164121218301286>. Citado na página 28.

BOJANOVA, I.; GALHARDO, C. E. C. Bug, fault, error, or weakness: Demystifying software security vulnerabilities. v. 25, n. 1, p. 7–12, 2023. ISSN 1941-045X. Conference Name: IT Professional. Disponível em: <https://ieeexplore.ieee.org/document/10077830>. Citado na página 33.

BUTTERSCOTCHAPART500. Q/A, *Creating a security program for a startup*. 2021. Disponível em: <https://www.reddit.com/r/cybersecurity/comments/the6zm/creating_a_security_program_for_a_startup/?rdt=39357>. Citado na página 62.

CANALS, S. Notícia, *Application security made simple*. 2024. Disponível em: <https://bebeez.eu/2024/05/06/application-security-made-simple-inside-belgian-startup-aikido-securitys-no-bs-security-platform-an>. Citado na página 62.

CARMEL. Time-to-completion in software package startups. In: *Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences HICSS-94*. IEEE Comput. Soc. Press, 1994. p. 498–507. ISBN 978-0-8186-5090-1. Disponível em: <http://ieeexplore.ieee.org/document/323468/>. Citado na página 31.

CROWNE, M. Why software product startups fail and what to do about it. evolution of software product development in startup companies. In: *IEEE International Engineering Management Conference*. IEEE, 2002. v. 1, p. 338–343. ISBN 978-0-7803-7385-3. Disponível em: <http://ieeexplore.ieee.org/document/1038454/>. Citado na página 32.

CRUZES, D. S.; DYBA, T. Recommended steps for thematic synthesis in software engineering. In: *2011 International Symposium on Empirical Software Engineering and Measurement*. IEEE, 2011. p. 275–284. ISBN 978-1-4577-2203-5 978-0-7695-4604-9. Disponível em: <http://ieeexplore.ieee.org/document/6092576/>. Citado na página 55.

CYBERTALENTS. Blog, *Top 10 Tips to Secure your Startup*. 2020. Disponível em: <https://cybertalents.com/blog/top-10-tips-to-secure-your-startup>. Citado na página 62.

DARE. *The Database of Abstracts of Reviews of Effects (DARE)*. 2002. York: University of York. NHS Centre for Reviews and Dissemination. Acessado em: 9 de nov. de 2023.

Disponível em: <https://www.crd.york.ac.uk/CRDWeb/AboutPage.asp>. Citado 2 vezes nas páginas 54 e 57.

DYBå, T.; DINGSøYR, T. Empirical studies of agile software development: A systematic review. v. 50, n. 9, p. 833–859, 2008. ISSN 09505849. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0950584908000256>. Citado 4 vezes nas páginas 21, 53, 54 e 57.

FASTERCAPITAL. artigo, *What you need to know about startup security*. 2024. Disponível em: <https://fastercapital.com/content/What-you-need-to-know-about-startup-security.html>. Citado na página 62.

FERDIANSYAH, D.; ISNANTO, R. R.; SUSENO, J. E. Strategy Indicators for Secure Software Development Lifecycle in Software Startups Based on Information Security Governance. *Journal of Internet Services and Information Security*, v. 13, n. 4, p. 104 – 113, 2023. Disponível em: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85178895532&doi=10.58346%2fJISIS.2023.I4.007&partnerID=40&md5=35de75ebe3fb5fb46505a1870bc2f9d6>. Citado na página 62.

GAAFAR, A. A.; FOUAD, K. M.; SADEK, M. A. Penetration Testing: A Cost-Benefit Analysis of Best Practices Implementation for Software Startups. In: *2024 6th Novel Intelligent and Leading Emerging Sciences Conference (NILES)*. [S.l.: s.n.], 2024. p. 242–245. Citado na página 62.

GAINES, J. Blog, *Build an information security program that scales with your startup*. 2024. Disponível em: <https://launchtn.org/build-information-security-program/>. Citado na página 62.

GAROUSI, V.; FELDERER, M.; MäNTYLä, M. V. Guidelines for including grey literature and conducting multivocal literature reviews in software engineering. v. 106, p. 101–121, 2019. ISSN 0950-5849. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0950584918301939>. Citado 11 vezes nas páginas 17, 19, 30, 44, 45, 46, 47, 51, 52, 54 e 57.

GIL, A. C. *Como elaborar projetos de pesquisa*. 6. ed. [S.l.]: Editora Atlas Ltda, 2017. ISBN 978-85-970129-2-7. Citado 2 vezes nas páginas 29 e 30.

HALELIUK, R. Blog, *Mental models for cybersecurity startup founders and why Microsoft cannot easily force nimble early-stage companies out of business*. 2023. Disponível em: <https://ventureinsecurity.net/p/mental-models-for-cybersecurity-startup>. Citado na página 62.

JAIN, P. Blog, *7 Ways to Make Sure Your Digital Startup More Secure*. 2024. Disponível em: <https://www.startupgrind.com/blog/7-ways-to-make-sure-your-digital-startup-more-secure/>. Citado na página 62.

JEBARA, A. V. artigo, *5 Steps to Put Your Startup On the Path Towards Better Security*. 2019. Disponível em: <https://www.seedstars.com/content-hub/learning-resources/5-steps-put-your-startup-path-towards-better-security/>. Citado na página 62.

K, P. Blog, *What Type of Software Testing Should a Startup Focus on First?* 2024. Disponível em: <https://testvox.com/what-type-of-software-testing-should-a-startup-focus-on-first/>. Citado na página 62.

KEKüLLüOğLU, D.; ACAR, Y. "we are a startup to the core": A qualitative interview study on the security and privacy development practices in turkish software startups. In: *2023 IEEE Symposium on Security and Privacy (SP)*. [S.l.: s.n.], 2023. p. 2015–2031. ISSN: 2375-1207. Citado 2 vezes nas páginas 28 e 62.

KELLER, G. Blog, *A Security Checklist for Your Startup*. 2021. Disponível em: <https://jumpcloud.com/blog/security-checklist-startup>. Citado na página 62.

KITCHENHAM, B. A.; BUDGEN, D.; BRERETON, P. *Evidence-based software engineering and systematic reviews*. [S.l.]: CRC Press, 2016. (Chapman & Hall/CRC innovations in software engineering and software development). OCLC: ocn932588149. ISBN 978-1-4822-2865-6. Citado 11 vezes nas páginas 21, 30, 44, 45, 46, 47, 50, 51, 53, 54 e 55.

KLOTINS, E.; UNTERKALMSTEINER, M.; GORSCHEK, T. Software engineering knowledge areas in startup companies: A mapping study. *arXiv.org*, Cornell University Library, arXiv.org, Ithaca, 2023. ISSN 2331-8422. Citado na página 28.

KOVALENKO, D. Blog, *Startup Software Development Guide (How We Build Successful Apps)*. 2024. Disponível em: <https://www.uptech.team/blog/software-development-for-startup>. Citado na página 62.

KRAVCENKO, V. Blog, *Security at Startup*. 2023. Disponível em: <https://vadimkravcenko.com/shorts/security-at-startup/>. Citado na página 62.

LANKA, P.; VAROL, C.; SHASHIDHAR, N. Strategies for a startup software-as-a-service organizations with minimal budget to achieve security and compliance goals. In: *2023 11th International Symposium on Digital Forensics and Security (ISDFS)*. [s.n.], 2023. p. 1–6. Disponível em: <https://ieeexplore.ieee.org/document/10131124>. Citado 4 vezes nas páginas 42, 43, 48 e 62.

LEFEBVRE, C.; MANHEIMER, E.; GLANVILLE, J. Searching for studies. *Cochrane handbook for systematic reviews of interventions: Cochrane book series*, Wiley Online Library, p. 95–150, 2008. Citado na página 30.

LI, Z. *et al.* Accomplishing more with less: The practice of cybersecure health technology design among danish startups. In: *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, 2023. p. 1–8. ISBN 978-1-4503-9422-2. Disponível em: <https://dl.acm.org/doi/10.1145/3544549.3585597>. Citado 2 vezes nas páginas 43 e 62.

MICROSOFT. *Microsoft Security Development Lifecycle (SDL)*. 2023. Página de informações. Acessado em: 29 de out. de 2023. Disponível em: <https://www.microsoft.com/en-us/securityengineering/sdl/practices>. Citado 3 vezes nas páginas 38, 39 e 40.

MOQOD. Blog, *Essentials of Secure Code Writing: Key Strategies for Startup Success | Moqod*. 2024. Disponível em: <https://moqod.com/blog/secure-code-writing>. Citado na página 62.

NEWS, T. F. Blog, *The essential cybersecurity tools every startup needs to stay secure — TFN*. 2024. Disponível em: <https://techfundingnews.com/the-essential-cybersecurity-tools-every-startup-needs-to-stay-secure/>. Citado na página 62.

NEWS, T. H. artigo, *Importance of Application Security and Customer Data Protection to a Startup*. 2021. Section: Article. Disponível em: <https://thehackernews.com/2021/01/importance-of-application-security-and.html>. Citado na página 62.

NICK, E. artigo, *How to Build a Robust Cybersecurity Strategy for Your Startup*. 2023. Disponível em: <https://www.datasciencecentral.com/how-to-build-a-robust-cybersecurity-strategy-for-your-startup/>. Citado na página 62.

NOVIKOV, I. artigo, *Building Cybersecurity Strong Defenses for Startups*. 2024. Disponível em: <https://www.wallarm.com/what/building-cybersecurity-strong-defenses-for-startups>. Citado na página 62.

O.J., N.; SWADIMATH, U. C. Start-ups by women in bengaluru. *Journal of international women's studies*, Bridgewater State College, Bridgewater, v. 22, n. 6, p. 22–35, 2021. ISSN 1539-8706. Citado na página 28.

OWASP, O. W. A. S. P. *Melhores Práticas de Codificação Segura OWASP*. 2022. Disponível em: <https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/>. Citado na página 36.

PARIZI, R. *et al.* How has design thinking being used and integrated into software development activities? a systematic mapping. v. 187, p. 111217, 2022. ISSN 01641212. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0164121222000024>. Citado na página 51.

PARSIFAL. Web Page, *Parsifal - Perform Systematic Literature Reviews*. 2025. Disponível em: <https://parsif.al/>. Citado na página 59.

PATERNOSTER, N. *et al.* Software development in startup companies: A systematic mapping study. v. 56, n. 10, p. 1200–1218, 2014. ISSN 09505849. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0950584914000950>. Citado na página 31.

RASHID, A. *et al. The Cyber Security Body of Knowledge v1.1.0*. University of Bristol, 2021. Version 1.1.0. Disponível em: <https://www.cybok.org/>. Citado 9 vezes nas páginas 33, 34, 35, 36, 37, 38, 39, 40 e 41.

RENDE, J. Revista, *Seven Cybersecurity Tips To Strengthen Your Startup's Security Posture*. 2024. Disponível em: <https://www.forbes.com/councils/forbestechcouncil/2024/06/04/seven-cybersecurity-tips-to-strengthen-your-startups-security-posture/>. Citado na página 62.

SAFECODE. *Fundamental Practices for Secure Software Development: Essential Elements of a Secure Development Lifecycle Program*. 2018. SAFECode Tech. Rep. Third Edition. Acessado em: 29 de out. de 2023. Disponível em: <https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf>. Citado 3 vezes nas páginas 17, 41 e 42.

SAGARD. *The Startup 7: Cybersecurity Fundamentals for Early Stage Startups*. 2020. Disponível em: <https://www.sagard.com/fr/the-startup-7-cybersecurity-fundamentals-for-early-stage-startups/>. Citado na página 62.

SCHLOPSNA, N. artigo, *Exploring the Quantum Computing Benefits on Startup Innovation and Security*. 2024. Disponível em: <https://www.spectup.com/resource-hub/impact-of-quantum-computing-on-startup-innovation-and-security>. Citado na página 62.

SHARMA, A. *Security at Early Stage Startup*. 2023. Disponível em: <https://www.youtube.com/watch?v=J7bGKx1E4BI>. Citado na página 62.

SOUZA, R.; CICO, O.; MACHADO, I. *A Survey on Software Engineering Practices in Brazilian Startups*. arXiv, 2021. Disponível em: <http://arxiv.org/abs/2108.00343>. Citado 3 vezes nas páginas 28, 29 e 48.

SOUZA, R.; MALTA, K.; ALMEIDA, E. S. D. Software engineering in startups: A single embedded case study. In: *2017 IEEE/ACM 1st International Workshop on Software Engineering for Startups (SoftStart)*. [S.l.: s.n.], 2017. p. 17–23. Citado 3 vezes nas páginas 28, 43 e 48.

SUNDAR, V. Blog, *11 Cybersecurity Tips Every Startup Should Follow*. 2017. Disponível em: <https://www.indusface.com/blog/everything-startups-need-know-security/>. Citado na página 62.

SUTTON, S. The role of process in software start-up. v. 17, n. 4, p. 33–39, 2000. ISSN 07407459. Disponível em: <http://ieeexplore.ieee.org/document/854066/>. Citado na página 31.

TEAM, D. artigo, *Basic Security Practices and Features for A Startup Software Product*. 2020. Disponível em: <https://devathon.com/blog/basic-security-practices-features-startup-software-product/>. Citado na página 62.

VAADATA. Blog, *What security needs and what type of pentest for a startup?* 2021. Disponível em: <https://www.vaadata.com/blog/what-security-needs-and-type-of-pentest-for-startup/>. Citado na página 62.

WANG, X. *et al.* Key challenges in software startups across life cycle stages. In: SHARP, H.; HALL, T. (Ed.). *Agile Processes, in Software Engineering, and Extreme Programming*. [S.l.]: Springer International Publishing, 2016. (Lecture Notes in Business Information Processing), p. 169–182. ISBN 978-3-319-33515-5. Citado 2 vezes nas páginas 31 e 32.

WHITAKER, T. Blog, *Startup Security: A Framework From Series B to F Funding*. 2023. Disponível em: <https://theporkskewer.medium.com/startup-security-a-framework-from-zero-to-100m-arr-6809e74e1b2a>. Citado na página 62.

WOHLIN, C. *Experimentation in Software Engineering: An Introduction*. Netherlands: Springer Nature, 2012. v. 6. (International Series in Software Engineering, v. 6). ISBN 1461546257. Citado na página 48.

WORKOS. Blog, *A Developer's Guide to Startup Security: 15 Ways to Secure Your Startup*. 2021. Disponível em: <https://workos.com/blog/a-developers-guide-to-startup-security-part-2>. Citado na página 62.

APÊNDICE A – Versão inicial *string* de busca

Tabela 11 – Aplicação da primeira versão da *string* de busca

Base	**String**	**Resultados**
IEEE Xplore	((("Document Title":"Startups"OR "Document Title":"startup") AND ("Document Title":"Agile Software Development"OR "Document Title" :"Development Practices"OR "Document Title":"Secure Software Development" OR "title:Security"OR "Document Title":"Software Engineering Practices"OR "Document Title":"Software Security Practices"OR "Document Title":"Software Security Approaches"OR "Document Title" :"Strategies"OR "Document Title":"Secure Software")) OR (("Abstract" :"Startups"OR "Abstract":"startup") AND ("Abstract":"Agile Software Development"OR "Abstract":"Development Practices"OR "Abstract":"Secure Software Development"OR "Abstract:Security"OR "Abstract":"Software Engineering Practices"OR "Abstract":"Software Security Practices"OR "Abstract":"Software Security Approaches"OR "Abstract":"Strategies"OR "Abstract":"Secure Software")) OR (("Author Keywords": "Startups"OR "Author Keywords":"startup") AND ("Author Keywords": "Agile Software Development"OR "Author Keywords":"Development Practices" OR "Author Keywords":"Secure Software Development"OR "Author Keywords": "Security" OR "Author Keywords":"Software Engineering Practices"OR "Author Keywords":"Software Security Practices"OR "Author Keywords": "Software Security Approaches"OR "Author Keywords": "Strategies"OR "Author Keywords":"Secure Software")))	195
ACM	Abstract:(("Startups"OR "startup") AND ("Agile Software Development"OR "Development Practices"OR "Secure Software Development"OR "Security"OR "Software Engineering Practices"OR "Software Security Practices" OR "Software Security Approaches"OR "Strategies"OR "Secure Software")) OR Title:(("Startups"OR "startup") AND ("Agile Software Development" OR "Development Practices"OR "Secure Software Development"OR "Security"OR "Software Engineering Practices"OR "Software Security Practices" OR "Software Security Approaches"OR "Strategies"OR "Secure Software")) OR Keyword:(("Startups"OR "startup") AND ("Agile Software Development" OR "Development Practices"OR "Secure Software Development"OR "Security"OR "Software Engineering Practices"OR "Software Security Practices" OR "Software Security Approaches"OR "Strategies"OR "Secure Software"))	89
SCOPUS	KEY (("Startups"OR "startup") AND ("Agile Software Development"OR "Development Practices"OR "Secure Software Development"OR "Security"OR "Software Engineering Practices"OR "Software Security Practices"OR "Software Security Approaches"OR "Strategies"OR "Secure Software")) OR TITLE (("Startups"OR "startup") AND ("Agile Software Development"OR "Development Practices"OR "Secure Software Development"OR "Security"OR "Software Engineering Practices"OR "Software Security Practices"OR "Software Security Approaches"OR "Strategies"OR "Secure Software")) OR ABS (("Startups"OR "startup") AND ("Agile Software Development"OR "Development Practices"OR "Secure Software Development"OR "Security"OR "Software Engineering Practices"OR "Software Security Practices"OR "Software Security Approaches"OR "Strategies"OR "Secure Software"))	3456

APÊNDICE B – Dados extraídos

Tabela 12: Extração dos Dados E1

Item alvo	**Valores**	**Página(s)**
Data da extração	20/01/2025	-
Título da publicação	"We are a startup to the core": A qualitative interview study on the security and privacy development practices in Turkish software startups	-
Data do estudo	01/05/2023	-
Autores	Dilara Keküllüoğlu Yasemin Acar	-
Base de dados	Scopus	-
Tipo de estudo	LF - Primário	-
Práticas de Segurança de Software	1. Alocar fundos para segurança 2. Contratar desenvolvedores com experiência em segurança 3. Criar metas e roadmaps para promover segurança 4. Estabelecer os papeis dos responsáveis pela segurança e privacidade. 5. Compartilhamento de bibliotecas terceiras seguras entre startups. 6. Desenvolvedores que discutem e compartilham entre si boas práticas de segurança criam softwares mais seguros.	-

Continuado na próxima página

Tabela 12: Extração dos Dados E1 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	No contexto do papel da indústria: 1- Alocar fundos para segurança desde o início do desenvolvimento. É um investimento a longo prazo. 2- Contratar desenvolvedores com experiência em aplicação de práticas de segurança em outras empresas. 3- Criar metas e roadmaps para promover segurança na empresa, realizar treinamentos e incentivar que os desenvolvedores apliquem o conhecimento adquirido. 4- Estabelecer os papeis dos responsáveis pela segurança e privacidade. 5- Compartilhamento de bibliotecas terceiras seguras entre startups. No contexto dos desenvolvedores: 1- Problemas de segurança e privacidade devem ser consideradas como problemas trataveis na modelagem do software, e não problemas que podem acontecer com os desenvolvedores. 2- Desenvolvedores que discutem e compartilham entre si boas práticas de segurança criam softwares mais seguros. No contexto dos educadores: 1- Segurança e Privacidade precisam ser abordados nos cursos de graduação.	-
Contexto de aplicação das Práticas de Segurança de Software	Startup de software turcas	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"São recomendações que podem melhorar a segurança, a privacidade e a usabilidade no desenvolvimento de software de startups turcas, bem como em startups em geral"	-

Continuado na próxima página

Tabela 12: Extração dos Dados E1 (Continuado)

Item alvo	**Valores**	**Página(s)**
Dificuldades encontradas	Orçamento limitado Conhecimento técnico Segurança não é algo pensado no design dos sistemas, só quando o sistema amadurece Priorização de requisitos de segurança quase que somente por força de regulamentações externas a empresa	-

Tabela 13: Extração dos Dados E2

Item alvo	**Valores**	**Página(s)**
Data da extração	13/01/2025	-
Título da publicação	11 Cybersecurity Tips Every Startup Should Follow	-
Data do estudo	25/04/2017	-
Autores	Venkatesh Sundar	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Ser proativo 2. Plano de segurança 3. Treinamento sobre ameaças 4. Atualização de sistemas 5. Configurar controle de acesso	-

Continuado na próxima página

Tabela 13: Extração dos Dados E2 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	1. Ser proativo "Seja diligente com a segurança da sua empresa e envolva toda a sua equipe no processo." 2. Plano de segurança Criação de um plano de como lidar com os diferentes ataques de segurança, disponível para todos os funcionários. Se qualquer ataque acontecer, melhorar o plano. 3. Treinamento sobre ameaças Manter empregados informados das melhores práticas de cyber segurança, ameaças recentes e usar multi-fatores de autenticação nas senhas. 4. Atualização de sistemas Manter os softwares atualizados quanto a patches de segurança, inclusive os sistemas usados pelos empregados. 5. Configurar controle de acesso Permitir acesso à dados de acordo com características do cargo.	-
Contexto de aplicação das Práticas de Segurança de Software	Startups de software, e pequenas e médias empresas dependentes de dispositivos e aplicativos móveis.	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Tornar a startup mais protegida de hackers	-
Dificuldades encontradas		-

Tabela 14: Extração dos Dados E5

Item alvo	**Valores**	**Página(s)**
Data da extração	13/01/2025	-
Título da publicação	A Developer's Guide to Startup Security: 15 Ways to Secure Your Startup	-
Data do estudo	05/08/2021	-
Autores	WorkOS	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Montar 'stack' de autenticação 2. Provisionar e desprovisionar usuários 3. Proteger as APIs 4. Configurar controle de acesso 5. "Patching"automático 6. Treinamento sobre ameaças 7. Capturar "logs"de auditoria 8. Escrever plano de segurança	-
Descrição das Práticas de Segurança de Software	1.Montar 'stack' de autenticação: Implementar autenticação multifator (MFA), single sign-on (SSO) e gerenciamento de senhas. 2.Provisionar e desprovisionar usuários: Sistema de onboarding e offboarding de usuários. 3.Proteger as APIs: Inventário de APIs, criptografia com TLS, revisão de exposição de dados, validação de inputs, firewall para APIs e rate limiting. 4.Configurar controle de acesso: Definir acesso a recursos e dados conforme o cargo ou atributos do empregado. 5."Patching"automático: Usar ferramentas para atualização automática assim que patches de correção estiverem disponíveis. 6.Treinamento sobre ameaças: Manter a empresa informada sobre ameaças e como se defender, com documentação e treinamentos escaláveis. 7.Capturar "logs"de auditoria: Armazenar e processar eventos estruturados para análise dos dados. 8.Escrever plano de segurança: Realizar revisões de segurança regulares e criar um plano de resposta a incidentes.	-

Continuado na próxima página

Tabela 14: Extração dos Dados E5 (Continuado)

Item alvo	**Valores**	**Página(s)**
Contexto de aplicação das Práticas de Segurança de Software	Startups de software em crescimento.	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"Quanto mais cedo você implementar essas medidas, mais forte será a infraestrutura de segurança duradoura da sua empresa".	-
Dificuldades encontradas		-

Tabela 15: Extração dos Dados E6

Item alvo	**Valores**	**Página(s)**
Data da extração	13/01/2025	-
Título da publicação	A Security Checklist for Your Startup	-
Data do estudo	16/07/2021	-
Autores	Greg Keller	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Configurar controle de acesso 2. Provisionar e desprovisionar usuários 3. Capturar "logs"de auditoria 4. Treinamento sobre ameaças 5. Seguir os "Zero Trust Security Principles"	-

Continuado na próxima página

Tabela 15: Extração dos Dados E6 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	1. Configurar controle de acesso Uso de autenticação multifator (MFA), single sign-on (SSO) e gerenciamento de senhas. 2. Provisionar e desprovisionar usuários Sistema de onboarding e offboarding de usuários. 3.Capturar "logs"de auditoria Armazenar e processar eventos estruturados para análise dos dados. 4. Treinamento sobre ameaças Manter a empresa informada sobre ameaças e como se defender, com documentação e treinamentos escaláveis. 5. Seguir os "Zero Trust Security Principles"Usar autenticação em todas as camadas do software, de forma que o acesso só é permitido se a identidade, o dispositivo e a rede são comprovadamente seguros por meio de multiplas camadas de segurança.	-
Contexto de aplicação das Práticas de Segurança de Software	Startups que possuem recursos escassos e pouco pessoal especializado em segurança.	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Espera-se que seguir as práticas descritas "... reduzirá drasticamente suas chances de uma violação, e simplesmente ter esses elementos implementados desmotivará a maioria dos atores maliciosos a tentar hackear sua organização."	-
Dificuldades encontradas		-

Tabela 16: Extração dos Dados E7

Item alvo	**Valores**	**Página(s)**
Data da extração	17/01/2025	-
Título da publicação	Accomplishing More With Less: The Practice of Cybersecure Health Technology Design Among Danish Startups	-
Data do estudo	23/04/2023	-
Autores	Ziru Li Nicola Dragoni Simon N. T. Vu Kevin Doherty	-
Base de dados	Scopus	-
Tipo de estudo	LF - Primário	-
Práticas de Segurança de Software	Promoção de conhecimento acerca de cybersegurança	-
Descrição das Práticas de Segurança de Software	A partir de um mix de abordangens de pesquisa com especialistas multidiciplinares e funcionários de uma healthtech dinarmaquesa uma ferramenta foi criada para "apoiar o desenvolvimento de tecnologias de saúde cada vez mais conscientes da segurança". A ferramenta funciona como uma "[...] autoavaliação baseada na web, através da qual o preenchimento de um pequeno questionário que abrange os objetivos dos utilizadores, o tempo disponível, as áreas de especialização e a confiança nos domínios da segurança cibernética, gera uma classificação das iniciativas de segurança cibernética recomendadas."	-
Contexto de aplicação das Práticas de Segurança de Software	Startups que desenvolvem soluções tecnológicas de saúde	-

Continuado na próxima página

Tabela 16: Extração dos Dados E7 (Continuado)

Item alvo	**Valores**	**Página(s)**
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"A solução fornece uma experiência interativa baseada nos principais desafios enfrentados pelas startups de tecnologia da saúde, incentiva ativamente os membros da equipe a melhorar continuamente a segurança cibernética de suas startups por meio de recursos gamificados, incluindo emblemas e tabelas de classificação, e fornece representação visual personalizada de iniciativas acionáveis com base nas necessidades individuais de startups."	-
Dificuldades encontradas	A utilização do modelo não garante a conformidade com padrões ISO ou IEC	-

Tabela 17: Extração dos Dados E9

Item alvo	**Valores**	**Página(s)**
Data da extração	14/01/2025	-
Título da publicação	Basic Security Practices and Features A Startup Software Product Should Have	-
Data do estudo	02/11/2020	-
Autores	Devathon Team	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Realizar testes de penetração 2. Desabilitar modo "debug"3. Implementar príncipio do Menor Privilégio 4. Escrever código seguro	-

Continuado na próxima página

Tabela 17: Extração dos Dados E9 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	1. Realizar testes de penetração Antes de lançar o software no mercado realizar testes de penetração na aplicação e na infraestrutura. 2. Desabilitar modo "debug"Desabilitar o modo debug do software ao lançar em produção 3. Implementar príncipio do Menor Privilégio O sistema deve executar apenas com as permissões necessárias, além de evitar conexões a internet desnecessárias. 4. Escrever código seguro Desenvolver código de maneira que esteja salvo de introdução de vulnerabilidades	-
Contexto de aplicação das Práticas de Segurança de Software	Startups em estágio inicial	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"As diretrizes acima ajudarão muito a manter a segurança dos produtos de software de sua startup rigorosa."	-
Dificuldades encontradas	Encontrar bibliotecas (códigos de terceiros) seguros	-

Tabela 18: Extração dos Dados E10

Item alvo	**Valores**	**Página(s)**
Data da extração	14/01/2025	-

Continuado na próxima página

Tabela 18: Extração dos Dados E10 (Continuado)

Item alvo	**Valores**	**Página(s)**
Título da publicação	Build an information security program that scales with your startup	-
Data do estudo	17/10/2024	-
Autores	Jeff Gaines	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Estabeler um time de InfoSec 2. Inventário e Gerenciamento de Ativos 3. Performar Avaliações de Risco e Escrever Plano de Segurança 4. Treinamento sobre ameaças 5. Gerenciar Riscos de Terceiros 6. Melhoria continua do Programa de InfoSec	-

Continuado na próxima página

Tabela 18: Extração dos Dados E10 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	1. Estabeler um time de InfoSec Time técnico e executivo que entende os riscos de ativos da empresa e conseguem guiar os esforços de segurança. 2. Inventário e Gerenciamento de Ativos Todo software, local e hardware que acessa, processa ou guarda dados críticos deve ser listado e gerenciado. 3. Performar Avaliações de Risco e Escrever Plano de Segurança Identificar ameaças e vulnerabilidades para cada ativo e desenvolver planos para mitigar, transferir ou aceitar os riscos. Além de um plano de ação caso seja atacado. 4. Treinamento sobre ameaças Treinar funcionários para as melhores práticas de segurança e a reconhecer ataques. Fornecer ferramentas de monitoramento. 5. Gerenciar Riscos de Terceiros Analisar práticas de seguranças de fornecedores e provedores que possuem acesso aos dados sensíveis da empresa. Usar contratos para reforçar a conformidade. 6. Melhoria continua do Programa de InfoSec Realizar avaliações e auditorias nas práticas do programa de InfoSec.	-
Contexto de aplicação das Práticas de Segurança de Software	Startups em crescimento que possuem como objetivo escalar de forma segura e sustentável	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Melhor defesa contra ataques de cybersegurança e enfrentar os riscos de segurança com confiança	-

Continuado na próxima página

Tabela 18: Extração dos Dados E10 (Continuado)

Item alvo	**Valores**	**Página(s)**
Dificuldades encontradas	Entender o porquê uma startup precisa de um programa de InfoSec. Implementar o programa pode ser desafiador, mas é necessário apenas começar.	-

Tabela 19: Extração dos Dados E11

Item alvo	**Valores**	**Página(s)**
Data da extração	14/01/2025	-
Título da publicação	Building Cybersecurity Strong Defenses For Startups	-
Data do estudo	26/02/2024	-
Autores	Ivan Novikov	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Sistema de Autenticação e Gerenciamento de Usuário 2. Isolamento da Infraestrutura 3. Gerenciamento Patch 4. Monitoramento e Backup 5. Configurar controle de acesso 6. Capturar "logs"de auditoria e centralizá-los	-

Continuado na próxima página

Tabela 19: Extração dos Dados E11 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	1. Sistema de Autenticação e Gerenciamento de Usuário Usar um sistema SSO (Single-Sign-On) e gerenciar como os usuários podem acessar os recursos da empresa 2. Isolamento da Infraestrutura Todo o sistema interno e os dados precisam ser protegidos por SSO ou por VPN. 3. Gerenciamento Patch Usar ferramentas de gerenciamento de versões e manter sempre o software passível de atualizações de segurança. 4. Monitoramento e Backup Monitorar serviços em produção e manter backups atualizados fora da infraestrutura principal. 5. Configurar controle de acesso Acesso a dados sensíveis apenas a pessoal autenticado e autorizado. 6. Capturar "logs"de auditoria e centralizá-los Manter logs centralizados fora da infraestrutura principal, para que seja possível correlacionar eventos, buscar funcionalidades e investigar problemas.	-
Contexto de aplicação das Práticas de Segurança de Software	Startups de Software	-

Continuado na próxima página

Tabela 19: Extração dos Dados E11 (Continuado)

Item alvo	**Valores**	**Página(s)**
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"Ao seguir essas etapas principais, as startups podem proteger seus negócios proativamente, fornecendo um ambiente seguro e protegido contra potenciais ameaças cibernéticas. Começar cedo, garantir processos de segurança claros, implementar apenas processos de missão crítica e não implementar mais de uma solução para cada processo são etapas essenciais no protocolo de segurança adequado para startups. [...]. Ao seguir essas etapas, as empresas podem esperar um futuro próspero no mundo moderno e consciente da segurança de hoje."	-
Dificuldades encontradas	"Entender quais aspectos da startup precisam estar seguros"	-

Tabela 20: Extração dos Dados E13

Item alvo	**Valores**	**Página(s)**
Data da extração	16/01/2025	-
Título da publicação	Essentials of Secure Code Writing: Key Strategies for Startup Success	-
Data do estudo	01/01/2024	-
Autores	Moqod	-
Base de dados	Google	-
Tipo de estudo	LC	-

Continuado na próxima página

Tabela 20: Extração dos Dados E13 (Continuado)

Item alvo	**Valores**	**Página(s)**
Práticas de Segurança de Software	1. Modelagem de Ameaças 2. Criar Segurança desde o dia um 3. Seguir Princípios de Codificação Segura 4.Usar Bibliotecas e Ferramentas Seguras 5. Testar o código continuamente 6. Adicionar testes automatizados na linha de desenvolvimento 7. Treinamento do time 8. Monitoramente e Auditoria de Logs	-
Descrição das Práticas de Segurança de Software	1. Modelagem de Ameaças Entender riscos e estrutar o sistema contra ameaças. 2. Criar Segurança desde o dia um Pensar sobre segurança desde o início do desenvolvimento, adicionando-a ao processo de desenvolvimento de software como um todo 3. Seguir Princípios de Codificação Segura Seguir os seguintes príncipios: controle de acesso; validação de dados de entrada das aplicações e criptografia de dados sensíveis. 4.Usar Bibliotecas e Ferramentas Seguras Usar ferramentas e bibliotecas constantemente atualizadas e que são mantidas por organizações confiáveis. 5. Testar o código continuamente Realizar análises dinâmicas e estáticas no código desenvolvido. 6. Adicionar testes automatizados na linha de desenvolvimento Configurar execução automática de testes em pipelines de integração e entrega contínua CI/CD. 7. Treinamento do time Manter time atualizado sobre práticas de segurança e lançamentos. 8. Monitoramente e Auditoria de Logs Monitorar sistemas em produção e revisar logs periodicamente	-
Contexto de aplicação das Práticas de Segurança de Software	Startups de software	-

Continuado na próxima página

Tabela 20: Extração dos Dados E13 (Continuado)

Item alvo	**Valores**	**Página(s)**
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Proteger a empresa de ataques como phishing, ransomware, e ataques DDoS	-
Dificuldades encontradas		-

Tabela 21: Extração dos Dados E14

Item alvo	**Valores**	**Página(s)**
Data da extração	16/01/2025	-
Título da publicação	Exploring the Quantum Computing Benefits on Startup Innovation and Securit	-
Data do estudo	28/03/2024	-
Autores	Niclas Schlopsna	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Uso de Criptografia Quântica 2. Uso de Ferramentas Quânticas	-
Descrição das Práticas de Segurança de Software	1. Uso de Criptografia Quântica Uso de algoritmos quânticos em abordagens de proteção de dados. Uso de computadores quânticos acelera a criptografia 2. Uso de Ferramentas Quânticas Uso de ferramentas quânticas para análise de risco	-

Continuado na próxima página

Tabela 21: Extração dos Dados E14 (Continuado)

Item alvo	**Valores**	**Página(s)**
Contexto de aplicação das Práticas de Segurança de Software	Startups	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	1. Uso de Criptografia Quântica Benefícios como: "Implementação de criptografia pós-quântica; sistemas de distribuição de chaves seguras; detecção de ameaças em tempo real usando soluções de computação quântica." 2. Uso de Ferramentas Quânticas "[...] startups que usam computação quântica para análise de risco reduziram suas taxas de erro em 65%."	-
Dificuldades encontradas	- Necessidade de Alto Investimento - Escassez de conhecimentos técnicos - Integração entre Sistemas Complexa - Acesso limitado a recursos - Retorno incerto de investimentos - Aumento do tempo de desenvolvimento - Mercado não conhece os benefícios da computação quântica	-

Tabela 22: Extração dos Dados E15

Item alvo	**Valores**	**Página(s)**
Data da extração	16/01/2025	-
Título da publicação	How to Build a Robust Cybersecurity Strategy for Your Startup	-
Data do estudo	21/02/2023	-
Autores	Edward Nick	-
Base de dados	Google	-

Continuado na próxima página

Tabela 22: Extração dos Dados E15 (Continuado)

Item alvo	**Valores**	**Página(s)**
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Entender o Cenário de Ameaças 2. Criar uma Política de Cyber Segurança 3. Controle de Acesso 4. Realizar Testes de Penetração 5. Treinamento do Time	-
Descrição das Práticas de Segurança de Software	1. Entender o Cenário de Ameaças Necessário entender os tipos de ataques, alvos comuns e como avaliar as vulnerabilidades da empresa 2. Criar uma Política de Cyber Segurança "Documento escrito que descreve as medidas que sua organização tomará para proteger informações e sistemas sensíveis de ataques cibernéticos. Ela deve incluir diretrizes para comportamento de funcionários, melhores práticas para gerenciamento seguro de dados e detalhes sobre como sua organização responderá a uma violação."3. Controle de Acesso Restringir acesso a dados sensíveis à cargos que necessitam dos dados 4. Realizar Testes de Penetração Encontrar fraquezas no sistema antes de um possível ataque 5. Treinamento do Time Gerar conhecimento para a equipe de engenharia acerca das vulnerabilidades, riscos e medidas preventivas	-
Contexto de aplicação das Práticas de Segurança de Software	Startups	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"[...] construção de uma estratégia de segurança cibernética do zero para proteger seus dados proprietários e garantir que seu negócio prospere nos próximos anos."	-

Continuado na próxima página

Tabela 22: Extração dos Dados E15 (Continuado)

Item alvo	**Valores**	**Página(s)**
Dificuldades encontradas		-

Tabela 23: Extração dos Dados E18

Item alvo	**Valores**	**Página(s)**
Data da extração	17/01/2025	-
Título da publicação	Penetration Testing: A Cost-Benefit Analysis of Best Practices Implementation for Software Startups	-
Data do estudo	19/10/2024	-
Autores	Ahmed Ali Gaafar Khaled M. Fouad Mohamed A. Sadek	-
Base de dados	IEEE Digital Library	-
Tipo de estudo	LF - Primário	-
Práticas de Segurança de Software	1. Controle de Acesso por Cargos 2. Validação Dados de Entrada 3. Criptografia de Dados Sensíveis 4. Manter pacotes externos atualizados 5. Implementação de uma Política de Segurança de Contéudo 6. Testes de Penetração	-

Continuado na próxima página

Tabela 23: Extração dos Dados E18 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	1. Controle de Acesso por Cargos Implementação de Robe-Based Access Control (RBAC) através de mecanismos de autenticação forte como JWT e OAuth 2. Validação Dados de Entrada Validar dados de entrada do usuário de acordo com contratos estabelecidos evitando ataques como SQL Injection e Cross-Site-Scripting (XSS) 3. Criptografia de Dados Sensíveis Evitar salvar dados sensíveis em armazenamento local ou cookies, optando por salvar em armazenamento de sessão seguro ou criptografando os dados. 4. Manter pacotes externos atualizados Uso do gerenciados de pacotes npm para manter dependências externas atualizadas contra vulnerabilidades conhecidas 5. Implementação de uma Política de Segurança de Contéudo "Content Security Policy (CSP) para restringir o carregamento de recursos e aumentar a segurança."6. Testes de Penetração Contratação de especialistas para execução de testes de penetração.	-
Contexto de aplicação das Práticas de Segurança de Software	Startup pequena com baixo número de clientes	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	O custo inicial de 7% do valor de negócio da empresa com a implementação de boas práticas de segurança foi significamente ultrapassado pelos benefícios ao longo do tempo. "Os benefíicios incluem impacto positivo no mercado e alta retenção de clientes, garantindo crescimento sustentado e estabilidade."	-

Continuado na próxima página

Tabela 23: Extração dos Dados E18 (Continuado)

Item alvo	**Valores**	**Página(s)**
Dificuldades encontradas	Custo inicial considerável para contratação de especialistas em testes de penetração e da implementação de boas práticas de segurança.	-

Tabela 24: Extração dos Dados E21

Item alvo	**Valores**	**Página(s)**
Data da extração	26/11/2023	-
Título da publicação	Security Evaluation by Arrogance: Saving Time and Money	-
Data do estudo	01/05/2017	-
Autores	Hussain M. J. Almohri Sayed A. Almohri	-
Base de dados	ACM Digital Library	-
Tipo de estudo	LF - Primário	-
Práticas de Segurança de Software	Avaliação de segurança por "arrogância".	-

Continuado na próxima página

Tabela 24: Extração dos Dados E21 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	"Esta fase de avaliação de segurança contou com a paixão e o desejo de membros arrogantes da equipe de liderar avaliações de segurança agressivas com a intenção de quebrar códigos escritos por outros". PG 12 Descrição do processo no qual a prática foi inserida: 1) Uma reunião conjunta das equipes de negócios e engenharia para determinar e concordar com os requisitos para a liberação. 2) A equipe de engenharia prepara um rascunho de requisitos e traduz os requisitos em tarefas que são publicadas em um sistema de colaboração online. 3) Os engenheiros começam a codificar e entregar os componentes ao coordenador de segurança. 4) O coordenador de segurança envia tarefas de segurança para hackers éticos externos e para o engenheiro de testes de segurança. 5) O coordenador de segurança recebe um relatório de possíveis vulnerabilidades na versão preliminar e convoca uma reunião com a equipe de engenharia. OBS (Etapa 6, 7 e 8 presente na caixa abaixo)	-

Continuado na próxima página

Tabela 24: Extração dos Dados E21 (Continuado)

Item alvo	**Valores**	**Página(s)**
Contexto de aplicação das Práticas de Segurança de Software	Uma startup de software que desenvolvia um sistema de pagamento mobile. 2012. Time de 4 engenheiros. Não sabiam como estava a qualidade do software lançado a cada release, tampouco se respeitavam as políticas de segurança dispostas aos clientes. Questionavam se seria possível uma "avaliação de segurança com custo quase zero". "Nossa hipótese era que um engenheiro arrogante com uma atitude tecnicamente negativa em relação aos colegas pode ser uma vantagem em vez de um fardo"PG 12 6) A equipe de engenharia decide um plano para remediar as vulnerabilidades e imediatamente começa a melhorar a versão. 7) O coordenador de segurança acompanha constantemente as melhorias no release junto à equipe de engenharia. 8) O coordenador de segurança confirma o lançamento (ou aprova um commit) quando o lançamento atinge um bom nível de segurança.	-

Continuado na próxima página

Tabela 24: Extração dos Dados E21 (Continuado)

Item alvo	**Valores**	**Página(s)**
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Sete vulnerabilidades importantes foram encontradas. PG 16 "Embora nosso processo possa parecer simplista, ele forneceu uma plataforma tranquila e flexível que atingiu nossos objetivos". PG 16 "Um resultado importante do nosso processo foi o custo insignificante do processo de avaliação da segurança, que dependia fortemente de factores humanos e de recursos reutilizados já atribuídos para outros fins". PG 16 "Embora tenhamos observado a paixão e o desejo pessoal de realizar avaliações de segurança, a eficácia dos fatores humanos e a motivação intrínseca de nossa equipe ainda precisam ser cientificamente estabelecidas". PG 16	-
Dificuldades encontradas	"Infelizmente, a literatura de pesquisa tem uma visão limitada de metodologias e diretrizes para avaliação de segurança de software desenvolvido em ambientes de startups". PG 12 "Além disso, para equipes de startups, pesquisar, aprender e aplicar corretamente um processo teórico como o Ciclo de Vida de Desenvolvimento de Segurança da Microsoft [5] pode não ser viável". PG 12 "A intensidade do desenvolvimento, as rápidas mudanças nos requisitos e a pressão sobre a equipe para gastar o mínimo de tempo no desenvolvimento de um componente, estavam entre os motivos que atrasaram um teste completo de segurança tanto das aplicações quanto dos serviços backend."PG 13 "houve falta de interesse em segurança na maioria dos outros engenheiros de software."PG 13	-

Tabela 25: Extração dos Dados E23

Item alvo	**Valores**	**Página(s)**
Data da extração	18/01/2025	-
Título da publicação	Startup DevSecOps Security	-
Data do estudo	09/12/2024	-
Autores	Anshu bansal	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Treinamento periodico de Segurança 2. Sistema de "Reporting"3. Testes de Penetração	-
Descrição das Práticas de Segurança de Software	Utilização de práticas de DevSecOps. "DevSecOps é uma filosofia de desenvolvimento que visa incorporar medidas de segurança na rotina padrão do DevOps." 1. Treinamento periodico de Segurança Realizar treinamento períodicos sobre segurança de software com time de desenvolvedores 2. Sistema de "Reporting"Criação de mecanismo para que qualquer funcionário possa relatar atividades suspeitas ou falhas de segurança não cobertas. 3. Testes de Penetração Realização de testes de penetração periodicamente.	-
Contexto de aplicação das Práticas de Segurança de Software	Startups	-

Continuado na próxima página

Tabela 25: Extração dos Dados E23 (Continuado)

Item alvo	**Valores**	**Página(s)**
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"Com práticas adequadas de segurança DevSecOps em vigor, sua equipe de desenvolvimento será versátil, sua empresa economizará dinheiro a longo prazo e seus clientes ficarão impressionados com a qualidade do software que você cria."	-
Dificuldades encontradas	Desenvolvedores não interessados nas práticas de DevSecOps	-

Tabela 26: Extração dos Dados E24

Item alvo	**Valores**	**Página(s)**
Data da extração	18/01/2025	-
Título da publicação	Startup Security: A Framework From Series B to F Funding	-
Data do estudo	21/08/2023	-
Autores	Tad Whitaker	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	1. Treinamento sobre Segurança 2. Programa de Testes de Penetração 3. Estabelecer Políticas 4. Processo de aceitação do questionário de segurança 5. Simulação de Resposta a Incidentes	-

Continuado na próxima página

Tabela 26: Extração dos Dados E24 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	1. Treinamento sobre Segurança Criar uma mentalidade de segurança em todas as ações de todos os funcionários de todas as áreas da empresa. 2. Programa de Testes de Penetração Validação de novas funcionalidades por profissionais externos 3. Estabelecer Políticas Política de resposta a incidentes, controle de acesso, backup e restauração, criptografia, gerenciamento de mudanças, violação de dados, segurança de funcionários, avaliação de risco de fornecedores, gerenciamento de vulnerabilidades, desastres para finanças, registro de auditoria, continuidade de negócios. 4. Processo de aceitação do questionário de segurança Estabelecer processo para que possíveis perguntas de clientes acerca da segurança do produto sejam respondidas. 5. Simulação de Resposta a Incidentes Reunir os times de engenharia trimestralmente para exercício de simulação de incidente, com o intuito de gerar um passo a passo de resposta imediata a essas situações.	-
Contexto de aplicação das Práticas de Segurança de Software	Startup com 50-150 funcionários Nenhum/Um/Alguns especialistas em segurança Clientes consideram o produto/serviço de alto risco dentro do modelo de ameaças Alguns funcionários possuem acesso a todos os dados sensíveis	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Estruturação de todo o contexto de segurança dentro da empresa	-

Continuado na próxima página

Tabela 26: Extração dos Dados E24 (Continuado)

Item alvo	**Valores**	**Página(s)**
Dificuldades encontradas	Alto custo	-

Tabela 27: Extração dos Dados E26

Item alvo	**Valores**	**Página(s)**
Data da extração	18/01/2025	-
Título da publicação	Strategies for a Startup Software-as-a-Service Organizations with Minimal Budget to Achieve Security and Compliance Goals	-
Data do estudo	01/05/2023	-
Autores	Phani Lanka Cihan Varol Narasimha Shashidhar	-
Base de dados	IEEE Digital Library	-
Tipo de estudo	LF - Primário	-
Práticas de Segurança de Software	1. Estabelecer Domínios e Objetivos de Segurança 2. Estabelecer Controles de Segurança e Responsável por Domínio 3. Estratégias Automatizadas 4. Validação de Conformidade Ontológica	-

Continuado na próxima página

Tabela 27: Extração dos Dados E26 (Continuado)

Item alvo	**Valores**	**Página(s)**
Descrição das Práticas de Segurança de Software	1. Estabelecer Domínios e Objetivos de Segurança Com base na aplicação e serviço desenvolvido a empresa deve definir os objetivos de segurança. Exemplos: Data Security; Secure SDLC; Software Config Security; Operating System Security; Equipment Security; Physical Security; Security Program Management; Asset Management; Incident Management; Change Management; Access Management. 2. Estabelecer Controles de Segurança e Responsável por Domínio Com base nos domínios e seus objetivos estabelecidos, controles de segurança são identificados para assegurá-los. Cada controle deve ter ao menos dois indivíduos, um responsável e outro para validar e verificar periodicamente, segregando as responsabilidades. continua a descrição no contexto.	-
Contexto de aplicação das Práticas de Segurança de Software	Startups de Software-as-a-Service 3. Estratégias Automatizadas Ciclos de execução e validação de controles de segurança devem ser realizados para melhorar a segurança. Durante a fase de design do sofware é importante que cada controle possa ser pensado em nível de implementação de código com microarquitetura para automatizá-lo, aproveitando os serviços do provedor em nuvem. 4. Validação de Conformidade Ontológica Criação de um documento de mapeamento global para mapear controles de segurança ao padrões de segurança da indústria. Esse documento permite a conformidade regulatória. Uma validação ontológica é proposta através de um fluxograma.	-

Continuado na próxima página

Tabela 27: Extração dos Dados E26 (Continuado)

Item alvo	**Valores**	**Página(s)**
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"[...] método inovador de implementação de segurança em startups e ajudamos a mantê-las seguras e atingir metas de conformidade." "Olhando para os preços, a implementação de segurança baseada em microarquitetura reduz significativamente os custos, fornece direção e ajuda na conformidade geral do serviço"	-
Dificuldades encontradas	"Não é possivel automatizar todos os controles de segurança dentro das tecnologias usadas na aplicação, requerindo processo manual de implementação." "A escolha da arquitetura que precisa ser implementada para um controle específico depende da facilidade de implementação, da complexidade do código e dos recursos da plataforma do provedor de serviços em nuvem"	-

Tabela 28: Extração dos Dados E27

Item alvo	**Valores**	**Página(s)**
Data da extração	20/01/2025	-
Título da publicação	Strategy Indicators for Secure Software Development Lifecycle in Software Startups Based on Information Security Governance	-
Data do estudo	30/11/2023	-
Autores	Doddy Ferdiansyah R. Rizal Isnanto Jatmiko E. Suseno	-
Base de dados	Scopus	-
Tipo de estudo	LF - Primário	-

Continuado na próxima página

Tabela 28: Extração dos Dados E27 (Continuado)

Item alvo	**Valores**	**Página(s)**
Práticas de Segurança de Software	Implementar SSDL (Secure Software Development Lifecycle)	-
Descrição das Práticas de Segurança de Software	Dentro de startups o papel dos CTOs e CEOs é extremamente vital para a implementação e successo do desenvolvimento seguro de software (SSDL) e da segurança do produto desenvolvido. "Portanto, para mitigar o risco de falha de startups de software resultante de produtos de software inseguros e não conformidade com práticas ou cultura de segurança no desenvolvimento de software, é necessário criar um roteiro de estratégia de segurança explícito: 1. Ambiente externo e contexto de negócios 2. Estrutura legal e regulatória e seu impacto 3. Mudanças em ameaças, vulnerabilidades, tecnologias e apetite ao risco 4. Cultura da empresa 5. Requisitos para alinhamento explícito com certas iniciativas ou estratégias de negócios."	-
Contexto de aplicação das Práticas de Segurança de Software	Startups de software	-

Continuado na próxima página

Tabela 28: Extração dos Dados E27 (Continuado)

Item alvo	**Valores**	**Página(s)**
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	"Uma estratégia abrangente para o ciclo de vida de desenvolvimento de software seguro em startups de software deve envolver fortemente o CEO e o CTO. Suas responsabilidades devem abranger aspectos-chave, como estabelecer políticas de segurança, alocar recursos para medidas de segurança, monitorar controles de segurança e promover uma cultura de segurança da informação dentro da organização." "Ao envolver a alta gerência, as startups de software podem priorizar a segurança da informação e integrá-la à sua estratégia geral de negócios."	-
Dificuldades encontradas	Na abordagem de governança do SSDL, os cargos de CEO e CTO são de muita responsabilidade e críticos no alcance da implementação do processo de desenvolvimento seguro de software, o que indica que se um deles falhar a segurança será muito comprometida.	-

Tabela 29: Extração dos Dados E29

Item alvo	**Valores**	**Página(s)**
Data da extração	18/01/2025	-
Título da publicação	The Startup 7: Cybersecurity Fundamentals for Early Stage Startups	-
Data do estudo	10/11/2020	-
Autores	Sagard	-
Base de dados	Google	-
Tipo de estudo	LC	-

Continuado na próxima página

Tabela 29: Extração dos Dados E29 (Continuado)

Item alvo	**Valores**	**Página(s)**
Práticas de Segurança de Software	1. Manuseio e Privacidade de Dados 2. Educação 3. Segurança em Nuvem 4. Autenticação do Usuário 5. "Patching"de Segurança 6. Desenvolvimento Seguro 7. Testes de Penetração	-
Descrição das Práticas de Segurança de Software	1. Manuseio e Privacidade de Dados Categorização de todos os dados sensíveis e que necessitam de controle de acesso por funcionários ou clientes 2. Educação Treinamento para toda a empresa acerca de segurança de software 3. Segurança em Nuvem Ao usar provedores de nuvem realizar configurações de segurança em todos os recursos usados 4. Autenticação do Usuário Usar ferramentas para autenticar usuários para que tenham acesso apenas ao que é devido 5. "Patching"de Segurança Garantir que dispositivos com conexão a internet estão atualizados com últimos patches de segurança 6. Desenvolvimento Seguro Adicionar segurança como requisito na arquitetura de aplicações. 7. Testes de Penetração Realizar testes de penetração.	-
Contexto de aplicação das Práticas de Segurança de Software	Startups em estágio inicial	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Defender startups de ameaças de cybersegurança.	-

Continuado na próxima página

Tabela 29: Extração dos Dados E29 (Continuado)

Item alvo	**Valores**	**Página(s)**
Dificuldades encontradas	Alocação de recursos	-

Tabela 30: Extração dos Dados E31

Item alvo	**Valores**	**Página(s)**
Data da extração	18/01/2025	-
Título da publicação	What security needs and what type of pentest for a startup?	-
Data do estudo	20/08/2021	-
Autores	Vaadata	-
Base de dados	Google	-
Tipo de estudo	LC	-
Práticas de Segurança de Software	Testes de Penetração	-
Descrição das Práticas de Segurança de Software	O objetivo do teste de penetração deve ser alinhado com o escopo em análise, assím como o relatório técnico entregue ao final do teste. A ocorrência dos testes deve estar alinhado com a velocidade do ciclo de entregas, sejam pequenas ou grandes. Os testes também devem levar em conta o aumento do risco de exposição da empresa, ou seja, conforme o número de clientes aumenta os requisitos de cybersegurança também aumentam, o que demanda testes que validem as possíveis vulnerabilidades.	-

Continuado na próxima página

Tabela 30: Extração dos Dados E31 (Continuado)

Item alvo	**Valores**	**Página(s)**
Contexto de aplicação das Práticas de Segurança de Software	Startups que já realizam ou querem realizar testes de penetração	-
Resultado encontrado (ou esperado) da aplicação das Práticas de Segurança de Software	Atender requisitos de segurança e requisições de clientes. Aumentar confiança dos clientes. Atender certificados de segurança.	-
Dificuldades encontradas	Adaptar o orçamento do cliente ao custo dos testes.	-