CENTRO PAULA SOUZA
FACULDADE DE TECNOLOGIA DE ARARAQUARA
CURSO DE GRADUAÇÃO EM SEGURANÇA DA INFORMAÇÃO

Título do Artigo: DESENVOLVIMENTO SEGURO: UTILIZAÇÃO DA MODELAGEM DE AMEAÇAS PARA IDENTIFICAÇÃO DE VULNERABILIDADES NA FASE DE DESIGN

Autores: Kethilyn Cristina Mota
kethilyn.mota@fatec.sp.gov.br

Sabrina Prestes de Oliveira
sabrina.oliveira25@fatec.sp.gov.br

Orientador: Fernando Tiosso
fernando.tiosso@fatec.sp.gov.br

Resumo

Este artigo analisa a função da modelagem de ameaças como instrumento estratégico para o desenvolvimento seguro de software, ressaltando sua importância na detecção e mitigação de vulnerabilidades desde as fases iniciais de um projeto. Em meio ao progresso tecnológico e o crescimento dos ataques cibernéticos, assegurar a proteção da informação se tornou essencial para manter a integridade, a confidencialidade e a disponibilidade dos sistemas. Assim, o artigo discorre sobre os conceitos fundamentais de segurança e dos frameworks STRIDE e DREAD, utilizados para organizar e priorizar riscos, além de fornecer um estudo de caso sobre uma falha relevante resultante da falta dessa prática preventiva. O estudo de caso demonstra como falhas não identificadas, como XSS e injeção de SQL, podem comprometer seriamente a segurança de um sistema e mostra que a implementação desses frameworks auxilia na redução de riscos e destaca a importância de integrá-los ao ciclo de desenvolvimento, inclusive em contextos ágeis, contribuindo para a tomada de decisões mais precisas e seguras e promovendo o desenvolvimento de sistemas mais robustos, resilientes e orientados à segurança desde sua concepção.

Palavras-chave: Modelagem de Ameaças; Desenvolvimento Seguro; Segurança de Software; Cibersegurança; Mitigação de Ameaças.

CENTRO PAULA SOUZA
FACULDADE DE TECNOLOGIA DE ARARAQUARA
CURSO DE GRADUAÇÃO EM SEGURANÇA DA INFORMAÇÃO

Abstract

This article analyzes the role of threat modeling as a strategic tool for secure software development, highlighting its importance in detecting and mitigating vulnerabilities from the early stages of a project. Amid technological progress and the growth of cyberattacks, ensuring information protection has become essential to maintaining the integrity, confidentiality, and availability of systems. Thus, the article discusses the fundamental concepts of security and the STRIDE and DREAD frameworks, used to organize and prioritize risks, in addition to providing a case study of a relevant failure resulting from the lack of this preventive practice. The case study demonstrates how unidentified flaws, such as XSS and SQL injection, can seriously compromise the security of a system and shows that the implementation of these frameworks helps reduce risks and highlights the importance of integrating them into the development cycle, including in agile contexts, contributing to more accurate and secure decision-making and promoting the development of more robust, resilient, and security-oriented systems from their conception.

Keywords: *Threat Modeling; Secure Development; Software Security; Cybersecurity; Threat Mitigation.*

Junho/2025

1. INTRODUÇÃO

Nos últimos anos, o desenvolvimento de software cresceu exponencialmente impulsionado pelo amplo acesso à internet e a diversidade de dispositivos como os *smartphones*. Nesse contexto, proteger os dados dos usuários tornou-se prioridade para empresas e governos e o desenvolvimento seguro de software despontou para garantir a disponibilidade dos sistemas e a confidencialidade e integridade das informações por eles armazenadas e trafegadas (Carneiro, 2023).

A criação de programas seguros visa salvaguardar informações confidenciais, prevenir acessos não autorizados, vazamentos de dados e garantir a privacidade dos usuários e a integridade dos sistemas, tornando-se cada vez mais necessário considerar esse tipo de implementação desde as etapas iniciais do software, de modo a promover um desenvolvimento mais resiliente e alinhado às exigências de proteção da informação (Red Hat, 2022).

Em meio as boas práticas do desenvolvimento seguro, existe a modelagem de ameaças, uma técnica que permite a identificação de potenciais riscos durante o desenvolvimento do produto possibilitando que eles sejam mitigados nas fases iniciais da sua produção.

Segundo Howard e Lipner (2006), essa abordagem é um passo fundamental para o desenvolvimento de softwares seguros, visto que ela é realizada durante a fase de projeto, permitindo as equipes detectarem possíveis falhas no sistema antes mesmo do seu desenvolvimento, economizando tempo, prevenindo aborrecimentos e reduzindo os gastos com correções futuras.

No entanto, a implementação de práticas como a essa pode parecer um desafio em um mundo cada vez mais ágil, pois exige tempo e análise detalhada de riscos, o que pode ser entendido como um obstáculo em relação a rapidez pretendida no ciclo de desenvolvimento de software. Contudo, um estudo sobre adoção da modelagem de ameaças em projetos ágeis evidenciou a

possibilidade de incorporar essa técnica de forma eficaz sem prejudicar a velocidade de desenvolvimento do projeto (Carneiro, 2023).

A maioria das vulnerabilidades de softwares exploradas por atacantes tem origem no início do desenvolvimento de software, pois é nesse momento que decisões cruciais, como a escolha de tecnologias, implementação de controles de segurança e definição de permissões de acesso, são tomadas.

Assim, com o propósito de explorar a utilização da modelagem de ameaças para identificação de vulnerabilidades na fase de design do software, o presente artigo foi desenvolvido em 5 (cinco) seções, sendo estas: a seção 2 (dois) dedicada à apresentação dos conceitos e fundamentos que embasam o estudo realizado, a seção 3 (três) que discorre sobre métodos e procedimentos utilizados para condução da pesquisa, a seção 4 (quatro) que evidencia a interpretação dos resultados e, por fim, a seção 5 (cinco) que apresenta as considerações finais.

2. REFERENCIAL TEÓRICO

Com a finalidade de vislumbrar o conhecimento inerente ao tema proposto, faz-se necessário entender o conceito de modelagem de ameaças, ferramentas que facilitam sua implementação, seus desafios, as boas práticas adotadas em sua utilização e algumas das terminologias utilizadas em todo seu contexto de aplicação.

2.1 Modelagem de ameaças

A modelagem de ameaças é uma prática fundamental na segurança de software, pois visa identificar, compreender e avaliar possíveis ameaças a um sistema antes mesmo que ele entre em operação.

De acordo com Yokoyama e Arima (2022), ela permite mapear os ativos mais importantes, definir o escopo do que será analisado incluindo os pontos vulneráveis e os possíveis caminhos que um atacante poderá explorar,

ajudando a mitigar riscos e a propor medidas preventivas desde as fases iniciais do desenvolvimento. Isso impede que vulnerabilidades se arrastem até estágios mais avançados do processo de desenvolvimento, onde seria mais difícil e custoso para corrigi-los.

A incorporação da modelagem de ameaças não apenas fortalece a segurança do sistema, mas também contribui para decisões mais assertivas quanto à arquitetura e aos aspectos críticos do projeto.

Ressalta-se que durante a utilização do software deve existir a revisão e atualização contínua do modelo de ameaças sempre que houver mudanças no sistema, garantindo que a segurança acompanhe sua evolução.

Com o intuito de facilitar a implantação da modelagem de ameaças, algumas ferramentas podem ser utilizadas como observa-se na seção 2.2.

2.2 Ferramentas que facilitam a implementação da modelagem de ameaças

Visando facilitar a implementação da modelagem de ameaças durante o desenvolvimento de um software, *frameworks* como STRIDE e DREAD podem ajudar a classificar e priorizar ameaças, além de facilitar a visualização de vulnerabilidades do sistema (Cavalcanti, 2024).

O *framework* STRIDE consiste em um modelo de modelagem de ameaças desenvolvido pela Microsoft que contribui para identificar diferentes tipos (categorias) de ameaças. O nome STRIDE é um acrônimo e cada tipo de ameaça é identificado da seguinte forma (Naik et al, 2024):

•*Spoofing*: Falsificação de identidade ou de informações;

•*Tampering*: Modificação de dados ou código;

•*Repudiation*: Ação de negar a autoria de uma transação ou evento.;

•*Information Disclosure*: Vazamento de dados confidenciais;

•*Denial of Service*: Impedir o acesso legítimo a recursos;

•*Elevation of Privilege*: Obter privilégios ou permissões que não deveriam ser concedidos;

Um dos principais benefícios do STRIDE é sua simplicidade, sendo amplamente adotado em ambientes de desenvolvimento de software para garantir que os sistemas sejam projetados com segurança desde o início e permitindo que desenvolvedores pensem sobre as ameaças durante a fase de design (Naik et al. 2024).

Após identificar e categorizar as ameaças, o *framework* DREAD ajuda a priorizá-las com base no impacto que podem causar, realizando uma avaliação quantitativa dos riscos e permitindo a identificação de quais delas devem ser tratadas com mais urgência. Conforme descrito por ALVES et al. (2024), o acrônimo DREAD significa:

•*Damage Potential*: Qual o impacto da ameaça caso ocorra;

•*Reproducibility*: Facilidade de replicar o ataque;

•*Exploitability*: Facilidade com que a ameaça pode ser explorada;

•*Affected Users*: Número de usuários que seriam impactados;

•*Discoverability*: Facilidade de identificar a ameaça;

Segundo Naik et al. (2024), os *frameworks* STRIDE e DREAD se complementam e fornecem uma abordagem integrada para priorizar ameaças durante o desenvolvimento, colaborando para a tomada de decisões de segurança mais assertivas desde as fases iniciais de desenvolvimento.

No entanto, modelagem de ameaças pode ser um trabalho um tanto quanto desafiador, como será mostrado na próxima seção.

2.3 Desafios apresentados na implementação da modelagem de ameaças

Implementar a modelagem de ameaças pode ser desafiador, mesmo sendo uma prática essencial para a segurança do software. Um dos maiores obstáculos é integrá-la às metodologias ágeis, que priorizam entregas rápidas. Muitas equipes enxergam esse processo como algo complexo e demorado, o que pode gerar resistência logo no início do desenvolvimento (Carneiro,2023).

Outro desafio, segundo Carneiro (2023), é a falta de profissionais especializados que podem contribuir para decisões equivocadas. Para aplicar a modelagem de ameaças de forma eficiente é preciso ter um conhecimento sólido tanto em segurança quanto em desenvolvimento, além do domínio das ferramentas adequadas que pode ser obtido por meio de treinamentos.

Por fim, manter a modelagem de ameaças atualizada ao longo do tempo não é algo simples, pois a medida que o sistema evolui, novas vulnerabilidades surgem, tornando-se essencial uma abordagem contínua. Em projetos longos e dinâmicos, essa adaptação pode se tornar um verdadeiro desafio (Carneiro,2023).

2.4 Boas práticas utilizadas na modelagem de ameaças

Para superar os desafios citados na seção 2.3 e tornar a modelagem de ameaças mais eficaz, algumas boas práticas podem ser adotadas. O primeiro passo é integrá-la de forma natural ao desenvolvimento ágil, sem comprometer a velocidade das entregas. Estudos, como os de Rafael Carneiro (2023), mostram que isso é possível com abordagens adaptadas, como sessões curtas e iterativas focadas nas funções mais críticas do sistema.

Outra estratégia essencial é o uso de ferramentas automatizadas, como as oferecidas pelo OWASP, que ajudam a identificar e simular ameaças com mais eficiência. Isso não apenas reduz a complexidade do processo, mas também acelera a detecção de vulnerabilidades. Para garantir o melhor uso dessas ferramentas, é fundamental que as equipes recebam treinamento contínuo em segurança da informação. Engenheiros de software, especialistas em segurança devem trabalhar juntos na identificação de riscos e soluções. Essa troca de conhecimento não só melhora a segurança do software, mas também garante que ele atenda às necessidades e expectativas do projeto (OWASP,2025).

Por fim, entender as terminologias comumente utilizadas no contexto das ameaças facilita o processo de entendimento, como se pode observar na seção 2.5.

2.5 Terminologias utilizados no contexto da modelagem de ameaças

Segundo Bruno Martins (2025), é essencial compreender terminologias utilizadas na modelagem de ameaças, especialmente para profissionais iniciantes na área, pois isso facilita o entendimento e aplicação das técnicas. Dentre as terminologias existentes, serão abordadas as que foram utilizadas no presente trabalho: XSS, *SQL Injection*, *Exploits*, *Insider* e CVEs.

Ataques de *Cross-Site Scripting* (XSS) envolvem a injeção de scripts maliciosos em sites legítimos, que são entregues ao navegador da vítima sem que este os reconheça como não confiáveis, permitindo o acesso de informações sensíveis, como cookies e tokens de sessão, podendo também ser usados para espalhar malware e realizar ataques de phishing (Kaspersky, 2025).

Já os ataques de *SQL Injection* permitem que hackers obtenham acesso ao banco de dados, comprometendo informações confidenciais que estão armazenadas. Não é raro que incidentes reportados na mídia envolvendo roubo de informações privadas sejam, na verdade, resultado de ataques de injeção de SQL bem-sucedidos. Os ataques de injeção de SQL não apenas figuram como uma das maiores ameaças à segurança de dados, mas também ocupam um lugar de destaque no Top 10 da *Open Web Application Security Project* (OWASP) que lista as vulnerabilidades mais críticas em aplicações web (SALEM FAKER, 2017).

Os *Exploits* permitem explorar falhas ou vulnerabilidades em sistemas e programas de computador, ou seja, podem ser usados por *hackers* mal-intencionados para causar danos ou roubar informações (Athena Security, 2025). Segundo Bui et al. (2025), também podem ser utilizadas por especialistas em segurança para testar e corrigir falhas.

Conforme explanado por Shaw, Ruby e Post (1998), um *Insider* é qualquer pessoa que tenha acesso legítimo aos sistemas e redes de computadores de uma organização e que pode causar danos consideráveis nesse ambiente, intencionalmente ou não, acessando informações sensíveis ou interrompendo as operações.

As Common Vulnerabilities and Exposures (CVEs) correspondem a falhas de segurança amplamente reconhecidas e documentadas, elas são organizadas de forma padronizada para facilitar o entendimento da comunidade sobre vulnerabilidades em diferentes tipos de software (HackerSec, 2023).

3. METODOLOGIA

Para o desenvolvimento deste trabalho foi necessário utilizar-se de duas estratégias metodológicas: pesquisa bibliográfica e análise do estudo de caso.

Segundo Gil (2019), a revisão bibliográfica é a base que sustenta qualquer pesquisa, visto que ela compreende o procedimento racional e sistemático que tem como objetivo proporcionar respostas aos problemas que são propostos, colocando o pesquisador em contato direto com tudo que foi escrito sobre o assunto e sustentando a outra estratégia metodológica que foi utilizada: análise do estudo de caso.

Já a análise do estudo de caso foi utilizada com o intuito de observar e analisar a aplicação prática dos conceitos estudados, pois segundo Yin (2015) o estudo de caso é uma estratégia de pesquisa que investiga conceitos dentro de um contexto da vida real, sendo útil para identificar se os conceitos foram bem definidos.

4. ANÁLISE E DISCUSSÃO DOS RESULTADOS

O estudo de caso apresentado nesta seção refere-se à startup TechAurora que estava crescendo rapidamente e decidiu lançar um novo

formulário de avaliação para coletar feedback de usuários sobre seus produtos. No entanto, pressionada pelo curto prazo, a equipe de desenvolvimento pulou a etapa de modelagem de ameaças e lançou a funcionalidade sem uma análise detalhada de segurança.

Em meio ao código, uma vulnerabilidade crítica foi ignorada e não foi detectada nos testes, tornando-se um grande risco: o formulário de coleta de dados permitia a injeção de código malicioso, possibilitando ataques de XSS e *SQL Injection*.

Aproveitando-se desse contexto, um atacante experiente analisou o sistema e criou um *exploit* para inserir um *script* malicioso no campo de comentário do formulário. Cada vez que um usuário com privilégio administrativo acessava as respostas desse formulário, o código malicioso executava um comando que roubava a sessão autenticada e enviava os *cookies* desta sessão para um servidor externo controlado pelo invasor, comprometendo contas administrativas com o objetivo de manipular as avaliações obtidas.

Aliado a isso, a ameaça se agravou quando um *Insider* da equipe de suporte, desavisado, compartilhou credenciais internas com terceiros, permitindo que os criminosos acessassem o painel de controle sem levantar suspeitas. Em poucos dias, vários clientes relataram que suas avaliações haviam sido alteradas ou excluídas sem autorização.

Quando a equipe de segurança da TechAurora identificou o problema, já havia ocorrido um impacto significativo na confiabilidade do sistema. A vulnerabilidade foi registrada como uma falha conhecida em um banco de dados CVE.

Contudo, o ataque serviu para evidenciar que negligenciar a modelagem de ameaças pode transformar um simples formulário de avaliação em uma porta de entrada para graves ataques. Como resposta, a TechAurora reforçou suas práticas de desenvolvimento seguro, garantindo que as funcionalidades do formulário de avaliação, bem como novas implementações, passassem por

revisões de segurança rigorosas antes de serem lançadas, aplicando *frameworks* como STRIDE e DREAD em suas análises.

4.1 Identificando ameaças com o STRIDE

Como fora descrito na seção 2.2, o *framework* STRIDE permitiu analisar ameaças de forma estruturada, classificando-as categorias. Para isso, foram elicitadas a sequência de operações para visualizar como as informações circulam dentro do sistema e onde podem existir vulnerabilidades, ajudando a identificar pontos críticos de segurança e definir estratégias para proteger cada etapa do fluxo de informações, assim obtendo-se as seguintes ações:

1. Usuário: Insere informações no formulário como nome, avaliação e comentário.
2. Formulário Web: Coleta e transmite os dados das avaliações ao servidor;
3. Servidor Web: Processa os dados recebidos das avaliações;
4. Sistema Gerenciador de Banco de Dados: Armazena os dados das avaliações no banco de dados;
5. Administração do Site: Garante o acesso e a gestão dos dados armazenados.

Dessa forma, a Tabela 1 descreve a aplicação do *framework* STRIDE em cada ameaça identificada no fluxo, definindo sua categoria, o componente do sistema afetado/impactado e uma possível medida de mitigação dos riscos.

Tabela 1 – Categorização das ameaças utilizando o STRIDE

Categoria	Ameaça Identificada	Componente Impactado	Medida de Mitigação
Spoofing	Envio de avaliações falsificando a identidade de um usuário legítimo.	Usuário	Implementar autenticação e reCAPTCHA

			para garantir a legitimidade do usuário
Tampering	Dados enviados pelo formulário podem ser adulterados antes de chegarem ao servidor.	Formulário Web, Servidor Web	Usar HTTPS para proteger a comunicação entre cliente e servidor.
Repudiation	O usuário pode alegar que não enviou determinada avaliação, dificultando a rastreabilidade.	Formulário Web	Implementar logs seguros e assinaturas digitais para auditoria de envio.
Information Disclosure	Dados das avaliações podem ser capturados e expostos.	Servidor Web, Banco de Dados	Criptografar dados sensíveis em trânsito e em repouso.
Denial of Service	Um atacante pode sobrecarregar o formulário com milhares de envios, interrompendo a execução do serviço.	Formulário Web	Implementar limites de envio e detecção de acessos maliciosos (rate limiting).
Elevation of Privilege	Um invasor pode explorar vulnerabilidades para acessar ou manipular dados no banco de dados.	Banco de Dados, Administração	Aplicar controle rigoroso de privilégios e validação no lado do servidor.

Fonte: Autores (2025)

Após a categorização das ameaças utilizando o modelo STRIDE, foi aplicado o *framework* DREAD visando priorizar a execução das atividades de correção mediante a gravidade de cada uma.

4.2 Priorizando os riscos com o DREAD

Conforme descrito por Willians ([s.d.]), descobrir vulnerabilidades é importante, mas ser capaz de estimar o risco associado ao negócio é igualmente importante. Dessa forma, o DREAD ajuda a medir o risco de cada ameaça ao avaliar cinco critérios principais como fora descrito na Seção 2.2. Para isso, cada ameaça recebe notas de 0 a 9, onde valores mais altos indicam maior gravidade. No final, a pontuação média desses critérios determina quais ameaças precisam de mais atenção.

Dessa forma, para a análise do estudo de caso foram estipulados esses os índices a seguir:

- 0 a <3: Baixo
- 3 a <6: Médio
- 6 a 9 : Alto

Assim, a Tabela 2 descreve a aplicação do DREAD em cada ameaça identificada no fluxo, definindo sua pontuação em cada critério.

Tabela 2: Pontuação das ameaças utilizando o DREAD

Ameaça Identificada	**Dano (D)**	**Reprodu-tibilidade (R)**	**Explo-ração (E)**	**Afeta Usuários (A)**	**Descobri-mento (D)**	**Média**
Avaliação Falsa (*Spoofing*)	5	7	6	6	6	6

Injeção de Código (XSS, *SQL Injection*)	8	7	8	9	8	8
Uso de credenciais comprometidas	7	6	7	8	7	7
Interceptação de Dados	6	6	5	7	6	6
Ataque de DoS	8	6	7	8	7	7.2
Elevação de Privilégios	9	8	8	9	8	8.4

Fonte: Autores (2025)

4.3 Ações tomadas para sanar as vulnerabilidades

Considerando a Tabela 2, as ameaças que apresentaram pontuação média superior a 7 foram tratadas com prioridade, por representarem riscos mais críticos ao ambiente analisado. A partir dessa análise, algumas ações corretivas foram aplicadas de forma imediata para mitigar essas vulnerabilidades. Abaixo, a descrição de cada uma delas:

1. **Elevação de Privilégios (Média: 8,4):** Essa foi a ameaça com a média maior, o que exigiu mais atenção. Para reduzir esse risco, foi adotado o princípio do privilégio mínimo, onde os acessos foram limitados somente ao necessário, foram reforçadas as permissões de usuários e serviços e monitorados os logs de atividades privilegiadas com mais rigor.
2. **Injeção de Código (*XSS*, *SQL Injection*) (Média: 8,0):** Para proteger o sistema contra ataques por injeção de códigos maliciosos, foram tomadas várias precauções. Primeiro, passou-se a verificar com mais cuidado tudo o que os usuários digitam no site, tal como informações em formulários e buscas. Também foi usada uma forma mais segura de conversar com o banco de dados, que impede que códigos maliciosos

sejam executados junto com as informações enviadas. Além disso, foi ativado um firewall de aplicação, que fica de olho no que entra e sai do sistema pela internet, e bloqueia automaticamente qualquer tentativa de invasão ou comportamento estranho.

3. **Ataque de DoS (*Denial of Service*) (Média: 7,2):** Para prevenir ataques de negação de serviço (DoS), foram limitadas as requisições por IP, evitando sobrecargas causadas por acessos excessivos. Também foram ativados alertas em tempo real para identificar tráfego anormal rapidamente. Além disso, a infraestrutura foi otimizada com uso de cache e balanceamento de carga, garantindo melhor desempenho e resistência contra esse tipo de ataque.
4. **Uso de credenciais comprometidas (Média: 7,0):** Considerando esse risco, foram fortalecidas as políticas de senha e passou-se a exigir autenticação multifator (MFA). Também foi implementada a expiração periódica de senhas e integradas ferramentas que ajudam a identificar possíveis vazamentos de credenciais.

As demais ameaças, que mostraram uma média inferior a 7 e classificadas como de prioridade média ou baixa, foram incluídas em um backlog de segurança e serão tratadas conforme o planejamento das próximas etapas de correção.

5. CONSIDERAÇÕES FINAIS

A pesquisa para a conclusão desse artigo teve como objetivo explorar a utilização da modelagem de ameaças como estratégia para mostrar como identificar vulnerabilidades do software na fase de design.

Com essa pesquisa, foi possível demonstrar de forma prática como fazer uma análise estruturada dos riscos e reforçar medidas de segurança logo no início do desenvolvimento com a utilização dos *frameworks* STRIDE e DREAD.

O estudo de caso foi criado para evidenciar que negligenciar a modelagem de ameaças pode gerar falhas críticas, comprometer a confiabilidade do sistema e gerar custos desnecessários com correções futuras.

Esse estudo contribui como um guia para o compliance de desenvolvimento seguro de softwares, visando mitigar vulnerabilidades e garantir a confidencialidade e disponibilidade do sistema, sustentando as diretrizes da Segurança da Informação, além de servir de base para o aprimoramento do conhecimento dos pesquisadores sobre o tema.

Os resultados obtidos indicam que a aplicação da modelagem de ameaças oferece vantagens diretas para profissionais e organizações. Para os programadores, ela apresenta um esquema definido para prever problemas desde a etapa de planejamento. Para as organizações, representa uma economia significativa ao prevenir retrabalho e possíveis vazamentos de informações, além de contribuir para a formação de uma cultura de desenvolvimento seguro, que é crucial para cumprir normativas como a LGPD.

Nesse contexto, segundo o Conselho Nacional de Justiça (2021), a conformidade com a LGPD e a ISO 27001 reforça ainda mais a necessidade de adotar práticas de segurança e proteção de dados em todas as fases de desenvolvimento do software. A implementação dessas normas garante que o tratamento de informações sensíveis seja realizado de forma ética e segura, promovendo maior segurança na entrega final. Portanto, integrar essas práticas ao processo de modelagem de ameaças contribui para a criação de sistemas mais robustos e alinhados às regulamentações de proteção da informação.

Quanto aos *frameworks* utilizados, apesar do DREAD ter se mostrado útil neste artigo, ele apresenta um nível de subjetividade nas análises, o que pode resultar em discrepâncias entre os avaliadores. Para pesquisas futuras, é sugerido o uso do *framework* CVSS (Sistema Comum de Pontuação de Vulnerabilidades), que é amplamente aceito para avaliar riscos de vulnerabilidades de forma mais objetiva e padronizada (NIST,2025).

No que diz respeito ao STRIDE, essa abordagem incentiva as equipes de desenvolvimento a adotarem uma postura preventiva, identificando e corrigindo

possíveis riscos e ameaças logo na fase inicial do projeto, tornando a segurança um elemento essencial. Para lidar com esses desafios, é necessário incentivar a formação contínua das equipes de desenvolvimento em segurança da informação, além de incorporar técnicas de modelagem de ameaças nos ciclos de vida do software (GOMES,2023). A utilização de ferramentas automatizadas para a análise de ameaças e para testes de segurança pode também minimizar o trabalho manual envolvido, tornando o processo mais eficiente e abrangente.

Logo, é fundamental que os profissionais compreendam a terminologia relacionada à modelagem de ameaças e dominem *frameworks* como STRIDE e DREAD, caso queiram integrar a segurança aos princípios de design de software, pois essa abordagem não só aprimora a defesa do sistema, mas também favorece um desenvolvimento mais ético e responsável.

REFERÊNCIAS BIBLIOGRÁFICAS

ALVES, Luis Fernando; AMALFI, Pedro; MARTIN, Santiago; SCHEPKE, Claudio; RODRIGUES, Elder; BERNARDINO, Maicon. Modelagem de Ameaças com STRIDE e DREAD: Uma Análise preliminar aplicada a um sistema IoT. In: ESCOLA REGIONAL DE ENGENHARIA DE SOFTWARE (ERES), 8., 2024, Santiago/RS. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2024. p. 218-227. Disponível em: https://doi.org/10.5753/eres.2024.4319. Acesso em: 03 mar. 2025.

ATHENA SECURITY. Empresa exposta a vulnerabilidades: hackers exploram agora!. Athena Security, 2025. Disponível em: https://athenasecurity.com.br/2025/05/07/empresa-exposta-vulnerabilidades-hackers-exploram-agora/. Acesso em: 30 maio 2025.

AZEVEDO, P. M. de; GIBERTONI , D. A importância do design centrado no usuário em metodologias ágeis como requisito de usabilidade. Revista Interface Tecnológica, [S. l.], v. 17, n. 2, p. 293–305, 2020. DOI: 10.31510/infa.v17i2.986. Disponível em: https://revista.fatectq.edu.br/interfacetecnologica/article/view/986. Acesso em: 21 out. 2024.

BUI, Quang-Cuong; IANNONE, Emanuele; CAMPORESE, Maria; HINRICHS, Torge; TONY, Catherine; TÓTH, László; PALOMBA, Fabio; HEGEDŰS, Péter; MASSACCI, Fabio; SCANDARIATO, Riccardo. A Systematic Literature Review on Automated Exploit and Security Test Generation. 2025. Disponível em: https://doi.org/10.48550/arXiv.2502.04953. Acesso em: 06 abr. 2025.

CARNEIRO, R. Adotando modelagem de ameaças em projetos ágeis de desenvolvimento de software. 2023. TCC (Graduação em Engenharia da Computação) – UFPE, Recife, 2023. Disponível em: https://repositorio.ufpe.br/handle/123456789/49925. Acesso em: 23 set. 2024.

CAVALCANTI, T. Modelagem de ameaças: desenvolvimento seguro. Amazon Kindle Direct Publishing, 2024. E-book.

CONSELHO NACIONAL DE JUSTIÇA (Brasil). Anexo V – Manual de Referência – Prevenção e Mitigação de Ameaças Cibernéticas e Confiança Digital. Brasília: CNJ, 2021. Disponível em: https://www.cnj.jus.br/wp-content/uploads/2021/03/AnexoVManualReferenciaPrevencaoMitigacaoDeAmeacasCiberneticasConfiancaDigitalRevisadoREV.docx.pdf. Acesso em: 6 abr. 2025.

GIL, A. C. Métodos e técnicas de pesquisa social. 6. ed. São Paulo: Atlas, 2019.

GOMES, Bruna. Importância do treinamento de segurança da informação nas empresas. Contacta, 27 jun. 2023. Disponível em: https://www.contacta.com.br/blog/importancia-do-treinamento-de-seguranca-da-informacao-nas-empresas. Acesso em: 06 abr. 2025.

HACKERSEC. O que são as CVE na área de cibersegurança. Disponível em: https://hackersec.com/o-que-sao-as-cve-na-area-de-ciberseguranca/. Acesso em: 06 abr. 2025.

HOWARD, M.; LIPNER, S. The security development lifecycle: a process for developing demonstrably more secure software. Redmond, WA: Microsoft Press, 2006.

ISO. The quest for cyber-trust. Publicado em 10 jan. 2019. Disponível em: https://www.iso.org/news/ref2359.html. Acesso em: 06 abr. 2025.

KASPERSKY. O que é um ataque de Cross-Site Scripting (XSS)? Kaspersky, 2023. Disponível em: https://www.kaspersky.com.br/resource-center/definitions/what-is-a-cross-site-scripting-attack. Acesso em: 04 abr. 2025.

MAUÉS, R. (s.d.). O que é modelagem de ameaças? Em Conviso AppSec. Glossário. Disponível em: https://www.convisoappsec.com/glossario/threatmodeling. Acesso em: 21 set. 2024.

MCGRAW, G. Software security: building security in. Boston, MA: Addison-Wesley, 2006.

NAIK, Nitin; JENKINS, Paul; GRACE, Paul; et al. A comparative analysis of threat modelling methods: STRIDE, DREAD, VAST, PASTA, OCTAVE, and LINDDUN. TechRxiv, 28 out. 2024. Disponível em: https://www.techrxiv.org/doi/full/10.36227/techrxiv.173014171.11449253/v1. Acesso em: 07 abr. 2025.

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (NIST). Métricas de vulnerabilidade do CVSS. Disponível em: https://nvd.nist.gov/vuln-metrics/cvss. Acesso em: 02 jun. 2025.

OWASP. Threat Modeling Cheat Sheet. Cheat Sheets Series, 2025. Disponível em: https://cheatsheetseries.owasp.org/cheatsheets/Threat_Modeling_Cheat_Sheet.html. Acesso em: 06 abr. 2025.

OWASP FOUNDATION. (s.d.). Open Web Application Security Project. Disponível em: https://www.owasp.org/. Acesso em: 20 set. 2024.

OWASP FOUNDATION. OWASP Threat Dragon: ferramenta de modelagem de ameaças, 2025. Tradução realizada com auxílio do Google. Disponível em: https://owasp.org/www-project-threat-dragon/. Acesso em: 07 abr. 2025.

PEEPLES, K. STRIDE Threat Model. DZone, 2 dez. 2015. Disponível em: https://dzone.com/articles/stride-threat-model. Acesso em: 24 set. 2024.

RED HAT. Segurança no ciclo de vida de desenvolvimento de software. Disponível em: https://www.redhat.com/pt-br/topics/security/software-development-lifecycle-security. Acesso em: 6 abr. 2025.

SALEM FAKER. A Systematic Literature Review on SQL Injection Attacks Techniques and Common Exploited Vulnerabilities. 2017. Tradução realizada com auxílio do Google. Disponível em: https://www.academia.edu/73369918/A_Systematic_Literature_Review_on_SQL_Injection_Attacks_Techniques_and_Common_Exploited_Vulnerabilities. Acesso em: 06 abr. 2025.

SHAW, Eric; RUBY, Keven G.; POST, Jerrold M. The Insider Threat to Information Systems. Security Awareness Bulletin, n. 2-98, jun. 1998. Disponível em: https://homes.cerias.purdue.edu/~mkr/sab.pdf. Acesso em: 06 abr. 2025.

SOMMERVILLE, I. Software engineering. 9. ed. Boston: Addison-Wesley, 2011. Capítulo 11. Disponível em: https://csunibo.github.io/ingegneria-delsoftware/libri/sommerville-software-engineering-9-ed.pdf. Acesso em: 24 set. 2024.

WILLIAMS, Jeff. OWASP Risk Rating Methodology. OWASP Foundation, s.d. Disponível em: https://owasp.org/www-community/OWASP_Risk_Rating_Methodology. Acesso em: 30 mar. 2025.

YIN, R. K. Estudo de caso: planejamento e métodos. 5. ed. Porto Alegre: Bookman, 2015.

YOKOYAMA, Rodrigo; ARIMA, Carlos Hideo. Modelagem de ameaça, análise de risco e suas aplicações na literatura. International Journal of Development Research, v. 12, n. 04, p. 55049-55055, abr. 2022. Disponível em: https://www.journalijdr.com/sites/default/files/issue-pdf/24250.pdf. Acesso em: 06 abr. 2025.