Associação Propagadora Esdeva
Centro Universitário Academia – UniAcademia
Curso de Bacharelado em Engenharia de Software
Trabalho de Conclusão de Curso – Artigo

INTEGRAÇÃO DA INTELIGÊNCIA DE AMEAÇAS NO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE: UMA ABORDAGEM PROATIVA PARA MITIGAR VULNERABILIDADES DESDE O INÍCIO

João Vitor Duque Cyrillo[1]
Centro Universitário Academia - UniAcademia, Juiz de Fora, MG
Romualdo Monteiro de Resende Costa[2]
Centro Universitário Academia - UniAcademia, Juiz de Fora, MG

Linha de Pesquisa: Engenharia de Software

RESUMO

A integração da Inteligência de Ameaças Cibernéticas (CTI) no Ciclo de Vida de Desenvolvimento de Software (SDLC), é explorada neste trabalho utilizando a abordagem *DevSecOps*. Visamos propor um *framework* teórico que fortaleça a segurança cibernética, equilibrando-a com a eficiência operacional e a qualidade do software. O estudo inicia com um mapeamento da literatura, abrangendo *DevSecOps*, a relevância da CTI e o estado atual da segurança no SDLC, identificando lacunas e a necessidade de uma segurança mais integrada e proativa. Focamos em identificar pontos para integrar a CTI no SDLC, desenvolvendo um *framework* adaptável e avaliando sua eficácia teórica. Esta análise envolve cada fase do SDLC, considerando a incorporação antecipada da CTI para mitigar riscos. O *framework* proposto é flexível, adequado para diversos contextos organizacionais, e sua eficácia é avaliada em comparação com práticas e modelos de segurança existentes. Este trabalho oferece contribuições acadêmicas e práticas, fornecendo insights sobre a segurança no desenvolvimento de software e um guia para implementar efetivamente a CTI no *DevSecOps*.

Palavras-chave: *DevSecOps*. Inteligência de Ameaças Cibernéticas. Ciclo de Vida de Desenvolvimento de Software. Segurança Cibernética. *framework* Adaptável.

ABSTRACT

[1] Discente do Curso de Sistemas de Informação do Centro Universitário Academia – UniAcademia.
[2] Docente do Curso de Sistemas de Informação do Centro Universitário Academia - UniAcademia

The integration of *Cyber threat intelligence* (CTI) into the Software Development Life Cycle (SDLC) is explored in this work using the *DevSecOps* approach. We aim to propose a theoretical *framework* that strengthens cybersecurity, balancing it with operational efficiency and software quality. The study begins with a literature review covering *DevSecOps*, the relevance of CTI, and the current state of security in the SDLC, identifying gaps and the need for more integrated and proactive security. We focus on identifying points to integrate CTI into the SDLC, developing an adaptable *framework*, and evaluating its theoretical effectiveness. This analysis involves each phase of the SDLC, considering the early incorporation of CTI to mitigate risks. The proposed *framework* is flexible, suitable for various organizational contexts, and its effectiveness is evaluated in comparison with existing security practices and models. This work offers both academic and practical contributions, providing insights into software development security and a guide for effectively implementing CTI in *DevSecOps*.

Keywords: *DevSecOps*. *Cyber threat intelligence*. Software Development Life Cycle. Cybersecurity. Adaptable *framework*.

1 INTRODUÇÃO

A indústria de tecnologia da informação tem vivenciado uma transformação significativa, sobretudo no desenvolvimento de software, impulsionada pela evolução das tecnologias digitais e a adoção de métodos ágeis, como o *DevOps,* definido como uma "metodologia de desenvolvimento que enfatiza a comunicação entre desenvolvedores de software e operações e visa tempos rápidos de entrega por meio de pipelines de entrega automatizados" (Jabbari et al., 2016, apud Koskinen, 2020).

O *DevOps*[3], que começou a tomar forma entre 2007 e 2008, levanta desafios significativos em segurança cibernética, dada a rápida mudança das ameaças digitais. Surge então, o *DevSecOps*, descrito como uma "mudança transformacional que incorpora cultura, práticas e ferramentas de segurança em cada fase do processo *DevOps*" (Deloitte, 2019, apud Koskinen, 2020).

[3] História do *DevOps*. Disponível em: <https://www.atlassian.com/br/*DevOps*/what-is-*DevOps*/history-of-*DevOps*>. Acesso em 15/06/2024

Motivado pela crescente complexidade e severidade das ameaças cibernéticas, o modelo *DevSecOps* representa uma solução promissora. A integração da Inteligência de Ameaças Cibernéticas (CTI) no processo *DevSecOps* é vista como uma estratégia essencial para antecipar e mitigar ameaças, alavancando uma postura de segurança mais proativa e preventiva.

Quanto à Inteligência de Ameaça Cibernética (*Cyber threat intelligence*), existem algumas variações para sua definição, todas giram em torno do conhecimento sobre ameaça e as ações derivadas desse conhecimento. Wiem Tounsi et al. (apud SILVA, 2023, p. 13) define Inteligência de Ameaça Cibernética (CTI) como o conhecimento baseado na análise de rastros que representam a ameaça, capaz de se traduzir em tomada de decisão.

O framework DTIIM (DevSecOps Threat Intelligence Integration Model) propõe a integração da CTI ao DevSecOps para desenvolver sistemas mais seguros e resilientes, oferecendo insights práticos e teóricos ao campo acadêmico do DevSecOps. A próxima seção mapeia a literatura sobre DevSecOps, CTI e o estado atual da segurança no SDLC. A metodologia e o desenvolvimento do DTIIM serão abordados na terceira seção. Os resultados da implementação, discussão e considerações finais, junto com sugestões para trabalhos futuros, serão apresentados na última seção.

2 REFERENCIAL TEÓRICO

Nesta Seção são descritos os conceitos e aspectos que fundamentam este trabalho, com destaque em *DevSecOps* (Seção 2.1), Inteligência de Ameaças Cibernéticas (Seção 2.2) e por fim o Estado Atual da Segurança no SDLC (Seção 2.3).

2.1 DEVSECOPS

O *DevSecOps* é uma evolução do *DevOps* que integra práticas de segurança ao ciclo de vida do desenvolvimento de software. Este conceito é baseado em princípios como "*Shift Left*", que é definido como “mover as práticas de segurança para as fases iniciais do desenvolvimento de software” (Koskinen, 2020), visando incorporar a segurança desde as fases iniciais do desenvolvimento. Além disso, a

automação e a integração contínua (CI/CD)[4] desempenham papéis cruciais, permitindo que testes de segurança sejam executados automaticamente em todas as etapas do pipeline de desenvolvimento.

Outro aspecto fundamental do *DevSecOps* abordado no *DTIIM* é a "*Security as Code*" (Segurança como código)[5], onde as práticas de segurança são automatizadas e integradas ao pipeline de desenvolvimento, permitindo a detecção e mitigação de vulnerabilidades em tempo real. Também é notório o conceito de "*Secure by Design*" (Seguro por Design)[6], onde as práticas de segurança são integradas desde a concepção inicial do sistema. Isso significa que a segurança é considerada como um requisito essencial desde o projeto inicial, garantindo que as decisões de arquitetura e design levem em conta as melhores práticas de segurança.

A adoção do DevSecOps envolve a criação de uma cultura de colaboração contínua entre as equipes de desenvolvimento, operações e segurança, promovendo uma abordagem integrada e holística para a segurança cibernética. A integração da segurança no DevOps é vista como uma mudança transformacional que inclui cultura, práticas e ferramentas de segurança em cada fase do processo DevOps (Deloitte, 2019, apud Koskinen, 2020).

2.2 INTELIGÊNCIA DE AMEAÇAS CIBERNÉTICAS (CTI)

Inteligência de ameaças é uma avaliação prospectiva baseada em evidências, incluindo contexto, implicações e conselhos orientados para a ação sobre uma ameaça ou vulnerabilidade de um sistema ou rede. Essa inteligência é produzida por meio da aplicação de métodos cognitivos individuais ou coletivos, e pode ser usada para informar decisões sobre a resposta a essa ameaça ou vulnerabilidade (apud SILVA, 2023, p. 13). A CTI[7] envolve as etapas de direcionamento, coleta, processamento, análise, disseminação de informações sobre

---

[4] O que é o CI/CD ? Disponível em: <https://www.redhat[.]com/pt-br/topics/*DevOps*/what-is-ci-cd>. Acesso em 15/06/2024

[5] Why implementing *Security as Code* is important for *DevSecOps* ? Disponível em: <https://about.gitlab[.]com/blog/2020/03/12/how-to-security-as-code/>. Acesso em 15/06/2024

[6] Secure-by-design: Which comes first, code or security? Disponível em: <https://securityintelligence[.]com/articles/secure-by-design-which-comes-first-code-or-securitry/>. Acesso em 15/06/2024

[7] As seis fases da inteligência contra ameaças cibernéticas. Disponível em: <https://leadcomm[.]com.br/2019/07/22/as-seis-fases-da-inteligencia-contra-ameacas-ciberneticas/>. Acesso em 15/06/2024

atividades maliciosas e feedback, permitindo que as organizações antecipem e respondam proativamente a ataques cibernéticos.

O processo de coleta contínua de dados de diversas fontes, como *feeds* de inteligência de ameaças[8], relatórios de vulnerabilidades, indicadores de comprometimento (IOCs)[9] e táticas, técnicas e procedimentos (TTPs)[10] dos atacantes, é fundamental para entender o panorama de ameaças e identificar riscos potenciais desde as fases iniciais do desenvolvimento.

A análise de ameaças identifica padrões de ataque, perfis de atacantes e vetores comuns, permitindo que as organizações priorizem as ameaças críticas e ajustem suas defesas. A disseminação de informações de inteligência garante que todos os *stakeholders* relevantes estejam cientes das ameaças atuais e das melhores práticas de mitigação.

A automação na CTI permite a coleta e análise eficiente de grandes volumes de dados. Ferramentas automatizadas, como sistemas de monitoramento e plataformas de inteligência de ameaças, identificam e correlacionam atividades maliciosas rapidamente, facilitando a resposta a incidentes. A CTI promove um ciclo de feedback contínuo, onde lições aprendidas aprimoram as práticas de segurança, incluindo revisões regulares dos planos de segurança e atualizações das medidas de mitigação com base em novas informações e análises.

2.3 ESTADO ATUAL DA SEGURANÇA NO SDLC

O Ciclo de Vida de Desenvolvimento de Software (SDLC) tradicionalmente segue um modelo linear, como o modelo cascata, onde a segurança é considerada apenas nas fases finais, durante o teste e a implementação. No entanto, essa abordagem tem se mostrado insuficiente diante do cenário atual de cibersegurança, onde as ameaças são cada vez mais sofisticadas e frequentes.

A abordagem *DevSecOps* proposta pelo *DTIIM* inclui a CTI em todas as fases do ciclo de vida, desde o planejamento até o monitoramento contínuo.

---

[8] *feeds* de inteligência de ameaça. Disponível em: <https://www.ibm.com/docs/pt-br/qradar-common?topic=dashboard-threat-intelligence-*feeds*>. Acesso em 15/06/2024

[9] Indicadores de Comprometimento. Disponível em: <https://www.gov.br/ctir/pt-br/assuntos/noticias/2021/indicadores-de-comprometimento>. Acesso em 15/06/2024

[10] <https[:]//cecyber[.]com/o-que-e-o-mitre-attck/.> Acesso em 15/06/2024

Durante a fase de planejamento, a análise de riscos[11] informada pela CTI é realizada para identificar potenciais ameaças desde o início do projeto. Esta análise é crucial para definir os controles de segurança necessários e estabelecer um plano de segurança que guiará o desenvolvimento do software.

Na fase posterior de análise de requisitos[12], a CTI é utilizada para garantir que os requisitos de segurança sejam identificados e incorporados desde o início. Através de dados de inteligência de ameaças os requisitos de segurança são continuamente atualizados para refletir as ameaças mais recentes e relevantes.

Durante a fase de design, a integração da CTI permite que a arquitetura de software[13] seja desenvolvida com resiliência contra ameaças conhecidas e emergentes. A modelagem de ameaças[14], informada pela CTI, é utilizada para identificar pontos fracos no design e ajustar a arquitetura conforme necessário.

Na fase de desenvolvimento, a CTI continua a desempenhar um papel vital, fornecendo insights contínuos sobre vulnerabilidades e ameaças emergentes que são incorporados ao desenvolvimento de código seguro. Ferramentas de análise de código estática (SAST)[15] e dinâmica (DAST)[16] são utilizadas para detectar e mitigar vulnerabilidades em tempo real, garantindo que o código seja desenvolvido de forma segura.

Durante a fase de teste, a CTI informa a realização de testes de segurança específicos, como testes de penetração e validação de IOCs e TTPs. Esses testes são cruciais para detectar vulnerabilidades, como *zero-days*[17] e CVEs[18] recentes,

---

[11] <https://www.devmedia.com.br/analise-de-riscos-e-o-teste-de-software/22831> Acesso em 15/06/2024

[12] <https://mercadoonlinedigital.com/blog/engenharia-de-requisitos/> Acesso em 15/06/2024

[13] <https://www.alura.com.br/artigos/padroes-arquiteturais-arquitetura-software-descomplicada> Acesso em 15/06/2024

[14] <https://www.cloudflare.com/pt-br/learning/security/glossary/what-is-threat-modeling/> Acesso em 15/06/2024

[15] <https://www.contacta.com.br/sast-dast-e-iast-entenda-a-diferenca-entre-esses-testes-de-vulnerabilidade-de-aplicativos/> Acesso em 15/06/2024

[16] <https://www.contacta.com.br/sast-dast-e-iast-entenda-a-diferenca-entre-esses-testes-de-vulnerabilidade-de-aplicativos/> Acesso em 15/06/2024

[17] <https://www.kaspersky.com.br/resource-center/definitions/zero-day-exploit> Acesso em 15/06/2024

[18] <https://blog.guardsi.com.br/o-que-sao-cves-entendendo-as-vulnerabilidades-e-exposicoes-comuns/> Acesso em 15/06/2024

que podem impactar o projeto. A realização de testes contínuos e a revisão do código garantem que todas as vulnerabilidades identificadas sejam mitigadas antes da implementação do software.

Na fase de implantação, a CTI assegura que todas as vulnerabilidades identificadas foram mitigadas e que a implantação é realizada de forma segura. O monitoramento contínuo é iniciado para detectar atividades suspeitas e responder a incidentes em tempo real. Ferramentas de automação são utilizadas para monitorar e correlacionar atividades maliciosas rapidamente, otimizando a resposta a incidentes.

Finalmente, na fase de operação e monitoramento, a CTI fornece informações cruciais para o monitoramento contínuo e a resposta a incidentes em tempo real. A implementação de programas de *Bug Bounty*[19] incentiva a descoberta de vulnerabilidades, proporcionando uma camada adicional de defesa. O ciclo de feedback contínuo, promovido pela CTI, assegura que as lições aprendidas com incidentes de segurança sejam usadas para aprimorar as práticas de segurança.

3 METODOLOGIA E DESENVOLVIMENTO DO FRAMEWORK

A estratégia de pesquisa do presente trabalho é qualitativa. Desta forma, primeiro se definiu o contexto do projeto e os insumos técnicos necessários. Após essa definição, foram realizadas buscas de referências para entender melhor os conceitos e suas nuances, utilizando as strings de busca: ("*threat intelligence*" OR "inteligência de ameaças") AND ("*DevSecOps*" OR "*development security operations*") AND ("*software developmen*t" OR "desenvolvimento de software") AND ("*cybersecurity*" OR "segurança cibernética") na base do *Google Scholar*. Após a coleta de dados, foi efetuada a etapa de análise.

O *DTIIM* é representado pela ilustração abaixo:

[19] <https://blog.bughunt.com.br/o-que-e-bug-bounty/> Acesso em 15/06/2024

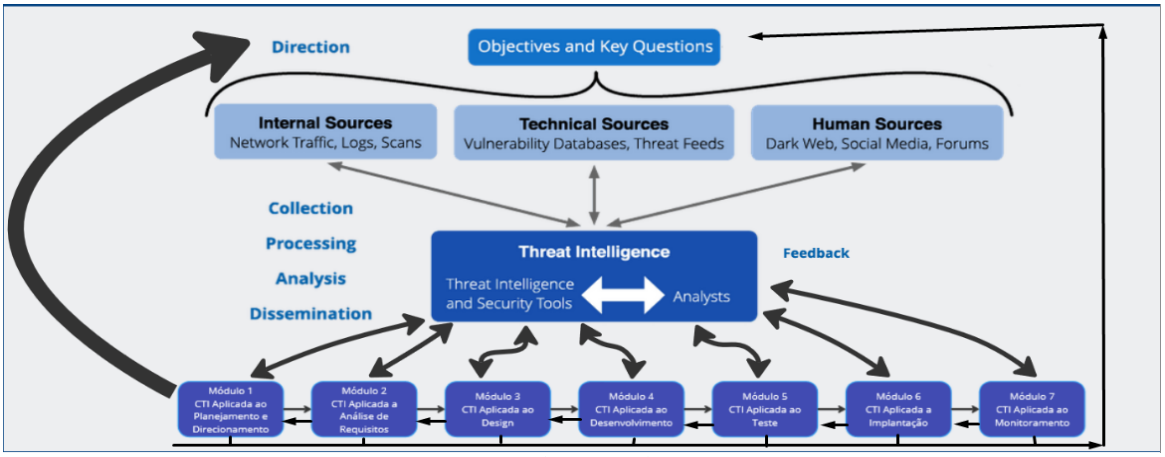

Figura 1: Ilustração do DTIIM

O *DTIIM* é estruturado em torno das fases do *DevSecOps*, com a integração de sete módulos-chave. Cada módulo corresponde a uma fase crítica do ciclo de vida do desenvolvimento de software, ajustado para refletir a integração da CTI em conjunto com as práticas de *DevSecOps*. Cada um dos módulos-chave apresenta quatro secções, representadas pelos inputs, outputs, processos e *stakeholders*. A seguir, seguem as definições referentes aos inputs, outputs, processos e *stakeholders*.

- Inputs: Os inputs indicam os dados, informações, recursos e materiais necessários para iniciar um processo ou atividade em um módulo específico.
- Outputs: Os outputs são os resultados ou produtos finais gerados a partir de um processo ou atividade dentro de um módulo.
- Processos: Os processos descrevem as etapas, atividades e fluxos de trabalho necessários para transformar os inputs em outputs.
- Stakeholders: Os *stakeholders* são os indivíduos, equipes ou funções que têm a responsabilidade de executar os processos e garantir a entrega dos outputs. No *framework*, esses *stakeholders* podem realizar uma ou mais funções (de acordo com a especificidade e necessidade do projeto).

Os níveis de maturidade do *DTIIM* são definidos como estágios que representam os requisitos mínimos que devem ser alcançados em cada módulo de implementação. Cada nível de maturidade possui como requisitos mínimos os inputs, processos e outputs de cada módulo para permitir a progressão ao próximo nível de implementação. Visando explicitar a adaptabilidade do *framework* aos diferentes contextos organizacionais, os *stakholders* não foram elencados como requisitos mínimos de implementação dos níveis de maturidade, focando assim nos processos, insumos e produtos finais gerados em cada módulo.

Nas secções posteriores serão abordados os módulos do *DTIIM*, iniciando pela secção “Módulo 1 – CTI Aplicada ao Planejamento e Direcionamento – Fase correspondente *DevSecOps* (Plan)” (Seção 3.1) até a secção final “Módulo 7 – CTI Aplicada ao Monitoramento - Fase correspondente *DevSecOps* (Operate, Monitor).” (Seção 3.7).

3.1 MÓDULO 1 – CTI APLICADA AO PLANEJAMENTO E DIRECIONAMENTO – FASE CORRESPONDENTE DEVSECOPS (PLAN)

Nesse nível 1 de maturidade, o *framework* busca realizar a integração da CTI desde o início do ciclo de desenvolvimento, estabelecendo uma base sólida para estratégias de segurança e a identificação precoce de ameaças potenciais.

- Checklist de Inputs Mínimos: Não há inputs nesse módulo.
- Checklist de Processos Mínimos:
  - ✓ Definição de Escopo e Objetivos: Neste processo, gerentes de projeto e executivos alinham o escopo e objetivos às metas organizacionais, com insights técnicos e operacionais das equipes de desenvolvimento, segurança e operações. Isso direciona a equipe de CTI e analistas de segurança nas áreas de foco, resultando no desenvolvimento do plano de coleta de dados de CTI.
  - ✓ Análise de Riscos informada por CTI: Este processo, guiado pelo plano de coletas de dados de CTI, avalia ameaças conhecidas e emergentes (*zero-days* e CVEs) utilizando bases de dados de vulnerabilidades, relatórios de inteligência e ferramentas de modelagem de ameaças para avaliações detalhadas de risco e integra feedback contínuo através de reuniões regulares com as equipes de desenvolvimento, segurança e operações para atualizar a análise de riscos.
  - ✓ Estabelecimento de Plano de Segurança: Neste processo, definem-se controles técnicos e organizacionais para mitigar riscos identificados, incluindo estratégias para responder a *zero-days* e ameaças emergentes com mecanismos de resposta rápida e planejamento de revisões de segurança periódicas e atualizações de CTI com base em um cronograma regular.
- Checklist de Outputs Mínimos:
  - ✓ Plano de Segurança: O plano de segurança clarifica os aspectos de segurança identificados específicos para o projeto.
  - ✓ Plano de Coleta de Dados de CTI: Este plano especifica o contexto, periodicidade e direcionamento da coleta de dados, focando em ameaças ao nicho de negócios da aplicação, tecnologias e componentes do projeto.

- ✓ Relatório de Análise de Riscos: O relatório de análise de riscos é o documento que lista as ameaças identificadas, suas probabilidades e impactos e as estratégias de mitigação propostas.

- Stakeholders:
  - ✓ Gerentes de Projeto.
  - ✓ Analistas de Segurança.
  - ✓ Equipe de CTI.
  - ✓ Executivos.
  - ✓ Representantes de Desenvolvimento.
  - ✓ Representantes de Segurança.
  - ✓ Representantes de Operações.

3.2 MÓDULO 2 – CTI APLICADA À ANÁLISE DE REQUISITOS – FASE CORRESPONDENTE DEVSECOPS (PLAN)

Nesse nível 2 de maturidade, o *framework* busca garantir que os requisitos de segurança sejam identificados e incorporados desde o início do projeto, utilizando insights de CTI para uma análise mais contextual e relevante aos objetivos do software.

- Checklist de Inputs Mínimos:
  - ✓ Plano de segurança.
  - ✓ Plano de coleta de dados de CTI.
  - ✓ Relatório de análise de riscos.
  - ✓ Feedback das equipes de desenvolvimento, segurança e operações: O feedback dessas são captados através de contribuições contínuas para garantir que os requisitos de segurança estejam atualizados e relevantes, em contraponto com suas perspectivas práticas de desenvolvimento, segurança, implantação e monitoramento.

- Checklist de Processos Mínimos:
  - ✓ Recepção do Plano de Segurança: Gerentes de projeto, analistas de requisitos e equipe de CTI utilizam o plano de segurança para atualizar os requisitos de segurança, garantindo que os riscos identificados no Módulo 1 sejam cobertos. Representantes das equipes de desenvolvimento, segurança e operações fornecem insights técnicos e operacionais para assegurar que os requisitos sejam práticos e implementáveis..
  - ✓ Incorporação de Dados de CTI: A equipe de CTI e analistas de segurança utilizam o plano de coleta de dados de inteligência para ajustar os requisitos de segurança, incorporando IOCs e TTPs. A identificação de ameaças emergentes é realizada através da análise contínua de fontes de CTI para ajustar os requisitos de segurança conforme necessário.

- ✓ Formulação e Documentação de Requisitos de Segurança: Os requisitos devem incluir a capacidade de resposta rápida a vulnerabilidades recém-descobertas, emergentes e *zero-days*.

- Checklist de Outputs Mínimos:
  - ✓ Requisitos de Segurança: Documento que especifica os padrões de segurança a serem seguidos pela equipe de desenvolvimento.
  - ✓ Diretrizes para a Equipe de Desenvolvimento: São especificações técnicas e procedimentos para garantir a implementação segura do software de acordo com os requisitos estabelecidos e identificados.

- *S*takeholders:
  - ✓ Gerentes de Projeto.
  - ✓ Analistas de Segurança.
  - ✓ Analistas de Requisitos.
  - ✓ Equipe de CTI.
  - ✓ Representantes de Desenvolvimento.
  - ✓ Representantes de Segurança.
  - ✓ Representantes de Operações.

3.3 MÓDULO 3 – CTI APLICADA AO DESIGN – FASE CORRESPONDENTE DEVSECOPS (PLAN)

Nesse Nível 3 de maturidade, o *framework* busca integrar o CTI na fase de design, assegurando que as arquiteturas de software sejam resilientes contra ameaças conhecidas e emergentes.

- Checklist de Inputs Mínimos:
  - ✓ Lista de Requisitos de Segurança.
  - ✓ Diretrizes para a Equipe de Desenvolvimento.
  - ✓ Dados de CTI Atualizados: Insights contínuos sobre ameaças emergentes e conhecidas fornecidos pela equipe de CTI.
  - ✓ Feedback das Equipes de Desenvolvimento, Segurança e Operações: São captados através de contribuições contínuas dessas equipes para garantir que a arquitetura e design do software estejam atualizadas e relevantes, em contraponto com suas perspectivas práticas de desenvolvimento, segurança, implantação e monitoramento.
- Checklist de Processos Mínimos:
  - ✓ Receber Requisitos de Segurança: Arquitetos e designers utilizam os requisitos de segurança para desenvolver uma arquitetura resiliente, com

feedback das equipes de desenvolvimento, segurança e operações sobre a praticidade e implementação desses requisitos.

- ✓ Avaliar e Ajustar Design com Base em CTI: A modelagem de ameaças pela equipe de CTI e analistas de segurança identifica pontos fracos no design e ajusta a arquitetura. A análise de vulnerabilidades incorpora dados de CTI, como relatórios de vulnerabilidades, IOCs e TTPs, para antecipar ataques e mitigar riscos específicos.
- ✓ Desenvolver Arquitetura de Software Segura: A implementação de controles de segurança é realizada estabelecendo controles técnicos, como WAF[20], além de controles organizacionais, como políticas de gerenciamento de patches. Esses controles são incorporados ao design para mitigar riscos identificados, além de fornecer um guia para equipe de desenvolvimento na implementação de uma arquitetura segura.
- ✓ Atualização Contínua do Design: A integração de novos dados de CTI é realizada continuamente no design com base em novas informações, ajustando a arquitetura para enfrentar novas ameaças emergentes, em conjunto com revisões periódicas do design.

- Checklist de Outputs Mínimos:
  - ✓ Arquitetura e Design de Software: Incorporam os controles de segurança identificados.
  - ✓ Documentação para Desenvolvimento: Especificações técnicas detalhadas para guiar a equipe de desenvolvimento na implementação prática de uma arquitetura segura.
- Stakeholders:
  - ✓ Arquitetos de Software.
  - ✓ Designers.
  - ✓ Analistas de Segurança.
  - ✓ Equipe de CTI.
  - ✓ Representantes de Desenvolvimento.
  - ✓ Representantes de Segurança.
  - ✓ Representantes de Operações.

[20] <https://www.cloudflare.com/pt-br/learning/ddos/glossary/web-application-firewall-waf/> Acesso em 15/06/2024

3.4 MÓDULO 4 – CTI APLICADA AO DESENVOLVIMENTO – FASE CORRESPONDENTE DEVSECOPS (CODE, BUILD)

Nesse Nível 4 de maturidade, o *framework* busca a implementação de práticas de codificação segura e revisão contínua do código para identificar e mitigar vulnerabilidades.

- Checklist de Inputs Mínimos:
  - ✓ Arquitetura e Design de Software.
  - ✓ Documentação para Desenvolvimento.
  - ✓ Dados de CTI Atualizados: Insights contínuos sobre vulnerabilidades e ameaças emergentes fornecidos pela equipe de CTI.
  - ✓ Feedback das Equipes de Desenvolvimento, Segurança e Operações: Feedback contínuo dessas equipes para garantir que o desenvolvimento esteja alinhado com as práticas de segurança e requisitos operacionais.
- Checklist de Processos Mínimos:
  - ✓ Receber Design Seguro: Os desenvolvedores utilizam a arquitetura e o design seguros para implementar o código de forma alinhada com os controles de segurança definidos e boas práticas de codificação. Os analistas de segurança revisam continuamente o código para garantir conformidade com os requisitos de segurança.
  - ✓ Desenvolver e Revisar Código com Práticas de CI/CD e *DevSecOps*: A implementação de controles de segurança no código é realizada através da aplicação de práticas de codificação segura e integração de ferramentas de análise de código estática (SAST), dinâmica (DAST) e de composição de software (SCA) entre outras relevantes. Testes proativos são incorporados na fase de desenvolvimento no pipeline CI/CD para reduzir o tempo de ajuste no código com base nos relatórios dos testes realizados. Revisões contínuas do código, incluindo peer reviews e testes automatizados, são realizadas para identificar e corrigir vulnerabilidades.
  - ✓ Atualização Contínua do Código: A integração de novos dados de CTI é realizada de forma contínua na etapa de desenvolvimento, atualizando o código com base em novas informações de CTI e ajustando as práticas de desenvolvimento para enfrentar novas ameaças identificadas.
- Checklist de Outputs Mínimos:

- ✓ Código de Software Seguro: Produto de software funcional e seguro, pronto para ser testado de forma complementar no módulo posterior.

- Stakeholders:
  - ✓ Desenvolvedores.
  - ✓ Equipe de CTI.
  - ✓ Analistas de Segurança.
  - ✓ Representantes de Desenvolvimento.
  - ✓ Representantes de Segurança.
  - ✓ Representantes de Operações.

3.5 MÓDULO 5 – CTI APLICADA AO TESTE – FASE CORRESPONDENTE DEVSECOPS (TEST)

Nesse Nível 5 de maturidade, a organização tem como objetivo a realização de testes de segurança específicos e contextuais, incluindo testes de penetração e análise de vulnerabilidades, informados pela CTI, para detectar vulnerabilidades conhecidas, *zero-days* e vulnerabilidades recentemente descobertas (CVEs) que possam impactar o projeto.

- Checklist de Inputs Mínimos:
  - ✓ Código de Software Seguro
  - ✓ Dados de CTI Atualizados: Informações sobre novas ameaças, IOCs (Indicadores de Comprometimento) e TTPs (Táticas, Técnicas e Procedimentos) fornecidas pela equipe de CTI.
  - ✓ Feedback das Equipes de Desenvolvimento, Segurança e Operações: O feedback dessas são captados através de contribuições contínuas na etapa de teste, em contraponto com suas perspectivas práticas de desenvolvimento, segurança, implantação e monitoramento.
- Checklist de Processos Mínimos:
  - ✓ Receber Código Desenvolvido: Os testadores e analistas de segurança avaliam e utilizam o código desenvolvido para iniciar a fase de testes de segurança.
  - ✓ Realizar Testes de Segurança Informados por CTI: Este processo inclui a realização de testes de penetração para identificar vulnerabilidades no software, como White Box Testing, Black Box Testing e Gray Box Testing, relevantes ao contexto do software. Além disso, a validação de IOCs e

TTPs envolve a criação de cenários em que esses elementos podem ser utilizados para comprometer um sistema, implementando simulações configuradas para usar esses IOCs, permitindo verificar se o sistema é capaz de bloquear ou alertar sobre essas tentativas em ambientes de teste ou sandboxes. A validação também inclui o uso de ferramentas de SIEM (Security Information and Event Management) para detectar atividades relacionadas a esses indicadores e TTPs.

  - ✓ Documentar e Analisar Resultados dos Testes: A documentação dos resultados dos testes de segurança inclui a identificação de vulnerabilidades, testes realizados e recomendações de mitigação.
- Checklist de Outputs Mínimos:
  - ✓ Relatórios de Testes: Documentam as vulnerabilidades identificadas, testes realizados e recomendações de mitigação.
  - ✓ Software Validado e Pronto para Implantação: Produto de software seguro e validado, pronto para ser implantado.
- Stakeholders:
  - ✓ Testadores.
  - ✓ Equipe de CTI.
  - ✓ Analistas de Segurança.
  - ✓ Representantes de Desenvolvedores.
  - ✓ Representantes de Segurança.
  - ✓ Representantes de Operações.

3.6 MÓDULO 6 – CTI APLICADA À IMPLANTAÇÃO – FASE CORRESPONDENTE DEVSECOPS (RELEASE, DEPLOY)

Nesse Nível 6 de maturidade, a organização tem como objetivo assegurar que as estratégias de segurança sejam adequadamente implementadas e testadas na fase de implantação do software.

- Checklist de Inputs Mínimos:
  - ✓ Relatórios de Testes.
  - ✓ Software Validado e Pronto para Implantação.
  - ✓ Dados de CTI Atualizados: Informações sobre novas ameaças e vulnerabilidades fornecidas pela equipe de CTI.

- ✓ Feedback das Equipes de Desenvolvimento, Segurança e Operações: Feedback contínuo dessas equipes para garantir que a implantação esteja alinhada com as práticas de segurança e requisitos operacionais.

- Checklist de Processos Mínimos:
  - ✓ Planejar a Implantação: As equipes de implantação e operações utilizam os relatórios de testes para planejar a implantação segura do software, assegurando que as vulnerabilidades identificadas foram mitigadas e que o plano de implantação inclui medidas de segurança adequadas. O planejamento de contingência envolve o desenvolvimento de planos para lidar com possíveis problemas durante a implantação, incluindo rollback de versões, escalonamento de problemas e notificações de segurança.
  - ✓ Implantação Segura: A execução do plano de implantação envolve a implementação do software em ambientes de produção de forma segura. Isso inclui a validação final dos controles de segurança e a realização de testes de fumaça para verificar se o software está funcionando conforme esperado. O monitoramento inicial é realizado para detectar e responder a possíveis problemas imediatamente, utilizando ferramentas de monitoramento para detectar atividades suspeitas ou anômalas.
  - ✓ Validação Pós-Implantação: A verificação de segurança é realizada para garantir que todas as medidas de segurança estão funcionando corretamente. A revisão de desempenho analisa se o software atende aos requisitos operacionais e de segurança, realizando testes de carga para garantir que o sistema pode suportar picos de tráfego sem falhar e que as políticas de segurança permanecem eficazes sob carga.
- Checklist de Outputs Mínimos:
  - ✓ Implantação Segura do Software: Software implantado em produção de forma segura.
  - ✓ Monitoramento Contínuo Pós-Implantação: Monitoramento contínuo para garantir a segurança e desempenho do software. Relatórios de monitoramento são gerados para fornecer visibilidade contínua sobre o estado de segurança do software.
- Stakeholders:
  - ✓ Operações.
  - ✓ Equipe de CTI.

- ✓ Analistas de Segurança.
- ✓ Representantes de Desenvolvimento.
- ✓ Representantes de Segurança.
- ✓ Representantes de Operações.

3.7 MÓDULO 7 – CTI APLICADA AO MONITORAMENTO – FASE CORRESPONDENTE DEVSECOPS (OPERATE, MONITOR)

No Nível 7, a organização tem como objetivo aplicar a CTI para um monitoramento contínuo do software em operação, identificando e respondendo a ameaças em tempo real e incentivando a identificação proativa de vulnerabilidades através de programas de *Bug Bounty*.

- Checklist de Inputs Mínimos:
  - ✓ Implantação Segura do Software.
  - ✓ Monitoramento Contínuo Pós-Implantação.
  - ✓ Dados de CTI Atualizados.
  - ✓ Feedback das Equipes de Desenvolvimento, Segurança e Operações: O feedback dessas equipes são captados através de contribuições contínuas na etapa de monitoramento, em contraponto com suas perspectivas práticas de desenvolvimento, segurança, implantação e monitoramento.
- Checklist de Processos Mínimos:
  - ✓ Monitorar Software em Operação: Ferramentas como o SIEM são utilizadas para detectar comportamentos anômalos e atividades suspeitas. Essa detecção pode ser realizada através da configuração de alertas para notificar a equipe de segurança em tempo real sobre atividades suspeitas, como tentativas de login falhadas repetidas ou acessos não autorizados a recursos críticos.
  - ✓ Analisar Dados de Monitoramento com CTI: A análise de dados de monitoramento é realizada através da utilização de dados de CTI para correlacionar eventos e identificar possíveis ameaças.
  - ✓ Responder a Incidentes: Inclusão de procedimentos de resposta rápida para mitigar ameaças identificadas, baseando-se em dados de CTI. A equipe de CTI analisa incidentes e fornece recomendações de mitigação, como bloquear IPs durante uma campanha de phishing e notificar usuários. Os planos de mitigação são continuamente ajustados e

atualizados após cada incidente, prevenindo futuras ocorrências. Por exemplo, uma análise forense pode isolar uma máquina afetada e aplicar patches de segurança após detectar uma intrusão.

- ✓ Gerar Relatórios: Os relatórios documentam atividades de monitoramento e incidentes detectados, fornecendo uma visão contínua do estado de segurança do software. Além desses documentos, há o compartilhamento de aprendizados com todas as equipes envolvidas, promovendo uma cultura de segurança e melhoria contínua. Por exemplo, relatórios semanais que detalham as atividades de monitoramento, incidentes detectados e as ações tomadas, compartilhados com a alta administração e as equipes de desenvolvimento para garantir transparência e cooperação.
- ✓ Implementar Programas de *Bug Bounty*: Incentivar a comunidade de segurança a identificar *zero-days* e outras vulnerabilidades críticas, oferecendo recompensas por descobertas dessas vulnerabilidades, fornecendo uma camada adicional de defesa e detecção proativa e preventiva, além de garantir que vulnerabilidades reportadas sejam validadas e corrigidas de forma ágil.

- Checklist de Outputs Mínimos:
  - ✓ Relatórios de Monitoramento: Documentam atividades de monitoramento e incidentes detectados.
  - ✓ Respostas a Incidentes: Descrevem as ações tomadas para mitigar ameaças.
  - ✓ Relatórios de *Bug Bounty*: Detalham as vulnerabilidades reportadas e as ações corretivas.
- Stakeholders:
  - ✓ Equipes de Monitoramento.
  - ✓ Analistas de Segurança.
  - ✓ Gestores do Programa de *Bug Bounty*.
  - ✓ Equipe de CTI.
  - ✓ Representantes de Desenvolvimento.
  - ✓ Representantes de Segurança.
  - ✓ Representantes de Operações.

Uma demonstração do trabalho encontra-se disponível no link: https://youtu.be/uDAUcPI82hk.

4 RESULTADOS E DISCUSSÃO

Utilizando o modelo BSIMM[21] (*Building Security In Maturity Model*) como referência teórica, foram analisadas e implementadas diversas práticas de segurança ao longo do ciclo de vida do desenvolvimento de software, divididas em quatro domínios principais: Governança, Inteligência, Pontos de Contato do SDLC e Implantação.

No domínio de Governança, implementamos políticas de segurança baseadas em análises de risco informadas por CTI, garantindo conformidade contínua com os requisitos de segurança. Na área de Inteligência, utilizamos modelos de ataque baseados em CTI para antecipar vetores de ataque, incorporamos recursos de segurança desde a fase de design e definimos requisitos de segurança específicos para o projeto, resultando em uma arquitetura de software mais resiliente.

Nos Pontos de Contato do SDLC, realizamos revisões de arquitetura com foco em segurança, utilizando dados de CTI, e integramos ferramentas automatizadas de análise de código (SAST e DAST) no pipeline CI/CD. Também realizamos testes de penetração e validação de IOCs e TTPs para identificar e corrigir vulnerabilidades críticas antes da implantação.

No domínio de Implantação, conduzimos testes de penetração informados por CTI, automatizamos patches e atualizações de segurança e implementamos monitoramento contínuo e resposta a incidentes em tempo real, garantindo a detecção e mitigação rápida de vulnerabilidades, mantendo a segurança contínua do software em produção.

Utilizando o modelo OWASP (Open Web Application Security Project) como referência teórica em relação ao DTIIM, observa-se que o OWASP oferece um amplo conjunto de projetos voltados para a segurança de aplicações, incluindo não apenas o OWASP Top Ten, que identifica e mitiga as 10 principais vulnerabilidades web, mas também ferramentas, guias e padrões como o OWASP SAMM (Software

[21] <https://www.inf.ed.ac.uk/teaching/courses/sp/2015/lecs/BSIMM6.pdf> Acesso em 15/06/2024

Assurance Maturity Model). Por outro lado, o DTIIM foca na integração da inteligência de ameaças ao DevSecOps para antecipar e mitigar ameaças cibernéticas, podendo realizar a combinação das abordagens para fortalecer a segurança. Ao incorporar as práticas e ferramentas do OWASP, o DTIIM enriquece sua metodologia, assegurando uma proteção mais abrangente e robusta contra ameaças cibernéticas.

A implementação do framework DTIIM demonstrou ser eficaz na integração da CTI com práticas DevSecOps, resultando em um software mais seguro e resiliente, além de contribuir para o campo acadêmico de DevSecOps, oferecendo insights práticos e teóricos essenciais para a segurança e confiabilidade dos sistemas digitais.

5 CONSIDERAÇÕES FINAIS

A abordagem integrada de segurança, desde as fases iniciais até o monitoramento contínuo, garante uma proteção abrangente contra ameaças emergentes. Para pesquisas futuras, é crucial explorar a aplicação prática do framework DTIIM em diferentes contextos organizacionais, como infraestrutura crítica, saúde e finanças, que têm requisitos de segurança críticos e específicos. Além disso, desenvolver metodologias para medir quantitativamente a eficácia da integração da CTI no DevSecOps pode proporcionar uma avaliação mais precisa dos benefícios dessa abordagem. Estudos comparativos entre diferentes modelos de maturidade de segurança e suas implementações práticas também podem fornecer insights adicionais para a evolução contínua das práticas de segurança no desenvolvimento de software.

6 REFERÊNCIAS

KOSKINEN, Anna. ***DevSecOps*: Building Security Into the Core of *DevOps***. Jyväskylä: University of Jyväskylä, 2019. Disponível em: https://jyx.jyu.fi/handle/123456789/67345. Acesso em: 17 jun. 2024.

DELOITTE. **Tech Trends 2019: Beyond the Digital Frontier**. Deloitte Insights 10th Anniversary Edition, 2019. Disponível em:

<https://www2.deloitte.com/content/dam/insights/us/articles/Tech-Trends-2019/DI_TechTrends2019.pdf>. Acesso em: 25 mai. 2019.

JABBARI, Ramtin et al. What is *DevOps*? A systematic mapping study on definitions and practices. In: **Proceedings of the scientific workshop proceedings of XP2016**. 2016. p. 1-11.

SILVA, R. M. **Proposta de um *framework* para melhoria da qualidade na produção de inteligência de ameaça cibernética.** 2023. Dissertação de Mestrado Profissional em Engenharia Elétrica, Universidade de Brasília, Brasília, 2023. Disponível em: http://repositorio2.unb.br/jspui/handle/10482/47957

Mohammed, N. M., Niazi, M., Alshayeb, M. & Mahmood, S. (2017). Exploring software security approaches in software development lifecycle: A systematic mapping study. Computer Standards & Interfaces, 50(2017), pp.107−115.