DASANAYAKE, S.D.L.V., SENANAYAKE, J. and WIJAYANAYAKE, W.M.J.I. 2025. DevSecOps implementation for continuous security in financial trading software application development. In *Proceedings of the 25th International conference on advanced research in computing 2025 (ICARC 2025): converging horizons: uniting disciplines in computing research through AI innovation, 19-20 February 2025, Belihuloya, Sri Lanka*. Piscataway: IEEE [online], pages 457-462. Available from: https://doi.org/10.1109/ICARC64760.2025.10963292

DevSecOps implementation for continuous security in financial trading software application development.

DASANAYAKE, S.D.L.V., SENANAYAKE, J. and WIJAYANAYAKE, W.M.J.I.

2025

DevSecOps Implementation for Continuous Security in Financial Trading Software Application Development

S.D.L.V. Dasanayake
Department of Industrial Management
Faculty of Science
University of Kelaniya
Sri Lanka
lashadyavidumini@gmail.com

Janaka Senanayake
Robert Gordon University
United Kingdom
University of Kelaniya
Sri Lanka
j.senanayake@rgu.ac.uk

W.M. J. I. Wijayanayake
Department of Industrial Management
Faculty of Science
University of Kelaniya
Kelaniya, Sri Lanka
janaka@kln.ac.lk

***Abstract*—DevSecOps incorporates security into the DevOps workflow, ensuring robust protection throughout the software development lifecycle. This research addresses the security gaps in financial trading applications, where traditional methods often prioritize speed over security. Using the Design Science Research Methodology (DSRM), the study examines secure coding practices, regulatory compliance, and incident response strategies. Findings highlight the benefits of embedding automated security testing and continuous monitoring to enhance resilience against evolving threats. Tailored developer training addresses knowledge gaps specific to trading platforms, ensuring compliance with regulatory demands and safeguarding sensitive financial data. By accelerating deployment timelines while strengthening security and compliance, this study demonstrates the critical role of a DevSecOps model in creating scalable, secure, and resilient trading applications.**

***Keywords*—*DevSecOps*, *cybersecurity*, *financial trading*, *secure coding*, *continuous monitoring*.**

I. Introduction

A. DevOps to DevSecOps

The integration of development, operations, and automation through DevOps has revolutionized software development processes, fostering collaboration and enhancing agility. However, traditional DevOps methodologies often treat security as a secondary concern, addressing it in the later stages of the Software Development Lifecycle (SDLC). This approach leaves software systems vulnerable to emerging cybersecurity threats, particularly in domains where data sensitivity and regulatory compliance are paramount, such as financial trading applications. DevSecOps extends the principles of DevOps by embedding security practices into every phase of the development lifecycle. It emphasizes proactive security measures, such as secure coding, continuous monitoring, and automated vulnerability assessment. This transition from DevOps to DevSecOps is essential for safeguarding financial trading platforms, which handle highly sensitive data and operate under stringent regulatory frameworks.

B. The Need of DevSecOps in Trading

Current practices in financial trading software development prioritize speed and agility over robust security measures. Security considerations are often deferred, leading to vulnerabilities that expose systems to significant threats, including data breaches, financial losses, and reputational damage. Developers' limited expertise in secure coding further exacerbates these challenges, as does the lack of standardized frameworks tailored for the unique requirements of trading platforms. Despite the critical need for security in these systems, the adoption of DevSecOps principles in financial trading applications remains inadequate.

C. Aim

This study aims to evaluate the effectiveness of integrating DevSecOps practices into the development of financial trading applications. Specifically, it focuses on creating a conceptual model to enhance security outcomes by examining the interplay of factors such as secure coding, team collaboration, and threat modeling.

The following research questions will be addressed in this study

Research Question 01

- How can security practices be effectively integrated and enhanced in the development lifecycle of financial trading applications?

Research Question 02

- What strategies can improve the adoption of secure development methodologies in financial trading software?

Research Question 03

- How can collaboration among development, security, and operations optimize security outcomes in financial trading applications?

II. Related Works

A. DevSecOps and Its Role in Secure Software Development

DevSecOps represents an evolution of the traditional DevOps framework by integrating security practices at every

stage of the software development lifecycle. Unlike traditional security approaches that address vulnerabilities post-development, DevSecOps advocates for proactive measures such as secure coding, automated vulnerability detection, and continuous monitoring. Researchers, [1] emphasize the benefits of DevSecOps, including the shift-left approach, which brings security considerations earlier in the development process, reducing overall risks and costs.

B. Security Practices in the Software Development Lifecycle (SDLC)

Traditional SDLC methodologies often treat security as an afterthought, resulting in vulnerabilities that are expensive and difficult to address in later stages. Secure Software Development Lifecycle (SSDLC) frameworks attempt to integrate security from the planning phase but are less dynamic compared to DevSecOps approaches. Challenges such as unclear guidelines, developer resistance, and limited automation have hindered the effective adoption of SSDLC practices [2].

DevSecOps enhances SDLC practices by embedding automated security testing, fostering collaboration across development, operations, and security teams, and aligning with industry standards such as OWASP and NIST [3]. Despite these advancements, the literature indicates a need for tailored frameworks to address specific security needs in trading platforms, where rapid deployment cycles and complex regulatory environments demand specialized solutions.

C. Security Challenges in Financial Trading Applications

Trading platforms operate in a unique environment characterized by high-frequency transactions, sensitive financial data, and stringent regulatory requirements. Studies have documented cybersecurity threats specific to this domain, including phishing, ransomware, and unauthorized breaches [4], [5]. The inherent risks associated with these platforms are exacerbated by the limited focus on security in traditional development workflows.

Most security frameworks focus on general financial applications and do not address the unique threats and vulnerabilities in trading platforms, and they lack adaptability to the dynamic nature of trading applications. This gap highlights the necessity of integrating DevSecOps principles tailored for trading platforms, ensuring the need of adaptability of models in a trading context.

D. Frameworks and Their Adaptation

Researchers adopts the ISSRM framework to categorize and address security risks, focusing on asset-related, risk-related, and risk treatment-related concepts [6].

Additionally, the study draws on previous work to highlight the importance of secure coding strategies, such as utilizing tools for static and dynamic code analysis, developer training, and gamified learning approaches [7], [8].While existing studies explore these techniques broadly, their application to trading platforms remains underexplored, representing a critical contribution of this research.

E. Threats in the Trading

Trading applications face various security threats that can severely impact their functionality and reliability. Key threats include data exfiltration and intellectual property theft, where sensitive financial data and proprietary algorithms are accessed or stolen, leading to financial loss and reputational damage [9], [10]. Market manipulation and insider trading further undermine trust in the system, with malicious actors influencing prices and misusing confidential information for personal gain [5], [11].

Cyberattacks, such as DDoS and ransomware, disrupt trading activities by overwhelming servers or encrypting critical data [12], [13]. Phishing tactics, including email, vishing, and smishing, also pose risks by deceiving users into revealing sensitive credentials [13]. Unpatched vulnerabilities and Advanced Persistent Threats (APTs) allow attackers to infiltrate systems undetected, while cryptocurrency platforms are at risk of hot wallet breaches and double-spending attacks [14], [15].

Non-compliance with KYC/AML regulations can result in legal penalties, and the lack of audit trails increases the risk of fraud [16], [17]. Fake news and disinformation campaigns also manipulate market behavior [10]. Lastly, poor network security and supply chain attacks expose platforms to unauthorized access and infiltration [9], [13].

F. Secure Coding Practices

Secure coding practices are essential for safeguarding trading applications, which handle sensitive financial data. Key practices include input validation and sanitization to prevent attacks like SQL injection and XSS, using whitelisting over blacklisting to allow only safe inputs, and sanitizing inputs to remove dangerous scripts [7], [18]. Output encoding is also critical, ensuring that data is securely processed in its intended environment to prevent script execution [19].

Authentication and authorization practices, such as enforcing strong password policies, implementing multi-factor authentication (MFA), and secure session management, help protect against unauthorized access [18]. For data protection, using encryption algorithms and avoiding hardcoding sensitive credentials are essential for safeguarding information [20], [11].

Effective error and exception handling should avoid exposing sensitive system details to users, while code reviews and automated tools ensure early detection of vulnerabilities [21], [18].

This study addresses several critical research gaps in the field of DevSecOps implementation for financial trading applications by measuring the overall security and resilience of these applications, examining the impact of collaboration between teams, secure coding practices, security frameworks, and threat modeling and risk assessment, along with the effects of trading software/project complexity and team experience and skill level.

III. Proposed Model

Despite the growing adoption of DevSecOps in software development, critical gaps persist, particularly within the context of financial trading applications. Existing frameworks are often generalized and fail to address the unique security requirements of trading platforms, which operate in high-stakes environments with sensitive data and stringent regulatory demands. Furthermore, there is insufficient focus on equipping developers with the secure coding skills necessary to meet the specialized needs of financial software. Additionally, gaps in automating compliance checks and integrating threat modeling into CI/CD pipelines hinder the ability to proactively manage vulnerabilities. Addressing these challenges, this study proposes a comprehensive DevSecOps model tailored to trading platforms, bridging these gaps and contributing to both theoretical understanding and practical advancements in financial cybersecurity.

identification, and risk management. The framework bridges critical components such as team collaboration, secure coding practices, and the adoption of security frameworks with risk analysis and mitigation strategies.

This framework emphasizes the interplay between independent variables, moderator variables, and their collective impact on the dependent variable—the overall security and resilience of financial trading applications.

By categorizing variables according to ISSRM's asset-related, risk-related, and risk treatment-related concepts, it ensures alignment with established risk management principles while tailoring its application to the unique challenges of financial trading systems. The inclusion of trading software/project complexity and team experience as moderating variables reflects the dynamic nature of security risk management, highlighting the need to consider project-specific complexities and team expertise in achieving robust security outcomes.

TABLE I
ISSRM Conceptual Areas, Variables, and Indicators

ISSRM Conceptual Area	**Variable**	**Indicators**
Asset-Related Concepts	Trading Software/Project Complexity	- Size of the codebase - Number of microservices/components - Number of external dependencies - Level of integration complexity - Types and volumes of transactions handled by the trading system
	Team Experience and Skill Level	- Years of experience in secure coding and DevSecOps - Number of security certifications held by team members - Past performance on similar projects - Training hours completed on security practices - Rate of compliance with secure coding practices
Risk-Related Concepts	Threat Modeling and Risk Assessment	- Accuracy of threat modeling - Documentation of risks and mitigations - Effectiveness of risk mitigation strategies
Risk Treatment-Related Concepts	Collaboration Between Teams	- Frequency of cross-functional meetings between development, security, and operations teams - Quality of communication tools - Involvement of security personnel in all stages of the SDLC - Level of integration of security tasks in DevOps pipelines - Perceived cooperation and trust between teams
	Secure Coding Practices	- Adherence to secure coding guidelines - Number of vulnerabilities detected during code reviews or scans - Developer training on secure coding (hours or certification levels)
	Security Frameworks	- Integration of security frameworks into CI/CD pipelines - Number of security tests and checkpoints within the framework - Effectiveness of framework implementation
-	Overall Security and Resilience of Financial Trading Applications (Dependent Variable)	- Mean time to detect/respond to incidents - Compliance with security standards - Frequency of successful attacks - Adoption of security best practices - User data protection rates - Resilience to emerging threats

A. Overview of the Conceptual Framework of the Model

The conceptual framework for this study provides a structured approach to understanding the factors that influence the security and resilience of financial trading applications. Grounded in the Information Systems Security Risk Management (ISSRM) framework, it integrates theoretical and practical elements to comprehensively address asset protection, risk

B. Model Validation

1) Data Collection: The study used a combination of primary and secondary data collection methods to investigate the subject matter and validate the proposed DevSecOps model.

- Primary Data Collection- Surveys were conducted to gather quantitative data from relevant stakeholders in

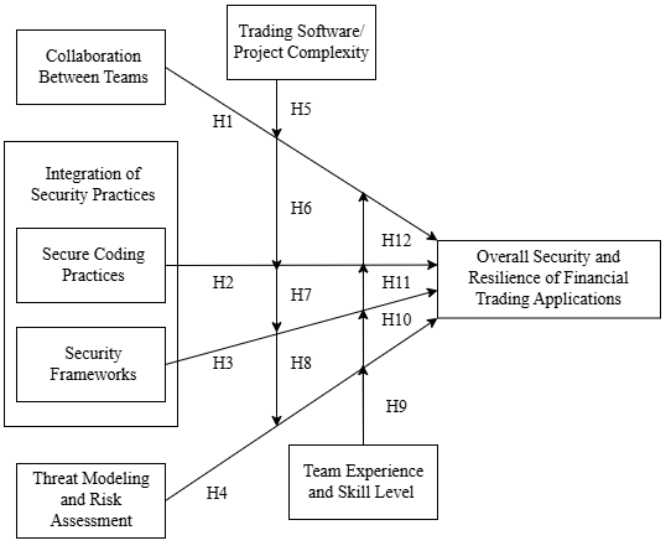

Fig. 1. Conceptual Framework Structure

financial trading software development. The target participants included software developers, IT managers, DevOps engineers, and security professionals. A structured questionnaire comprising demographic questions and items mapped to the conceptual framework was used, employing Likert scales to quantify perceptions and practices.

- Secondary Data Collection- An extensive literature review provided foundational insights, identifying current trends, security gaps, and best practices in DevSecOps implementation for trading applications. Sources included peer-reviewed journal articles, conference proceedings, and industry reports.
- Pilot Testing- Prior to distribution, the survey underwent pilot testing with four software engineering professionals, including two from trading software development firms, to improve question clarity and response usability.

2) Data Analysis: The data analysis was performed using Partial Least Squares Structural Equation Modeling (PLS-SEM) with SmartPLS 4.0.

- Preliminary Data Analysis Before conducting detailed statistical tests, the dataset was analyzed using SPSS software, and the following steps were performed:
  - All survey questions were made mandatory, minimizing the occurrence of missing data. Outliers were identified and addressed to mitigate their impact on the dataset and prevent skewed results. Normality assessments were conducted to confirm that the data distributions met the assumptions required for PLS-SEM analysis. Additionally, a demographic analysis was performed, providing descriptive statistics to profile respondents, including their roles, experience levels and education levels.

3) Measurement Model Assessment:

- Formative Variables - The formative constructs in this study were Collaboration Between Teams (CBT), Project Complexity (PC), and Overall Security and Resilience (OSR). These constructs were evaluated for collinearity using Variance Inflation Factor (VIF), with all values below 5, confirming no multicollinearity. The significance and relevance of indicators were assessed through bootstrapping, with significant weights indicating meaningful contributions. Convergent validity was confirmed, as correlations between the constructs and their global measures exceeded 0.7.
- Reflective Variables The reflective constructs were Team Expertise and Skills (TES) and Threat Modeling and Risk Assessment (TMR). Internal consistency reliability was assessed using Cronbach's Alpha and Composite Reliability (CR), both exceeding 0.7. Convergent validity was confirmed, with all outer loadings above 0.7 and Average Variance Extracted (AVE) values above 0.5. Discriminant validity was ensured through cross-loadings, the Fornell-Larcker Criterion, and the Heterotrait-Monotrait Ratio (HTMT), which was below 0.85 for all constructs.

4) Moderation Assessment:

- The moderation assessment examined how moderator variables influenced the relationships between independent and dependent variables. Interaction terms, combining independent and moderator variables, were incorporated into the structural model to evaluate moderation effects. Path coefficients, derived through bootstrapping, provided p-values and coefficients to quantify the significance and strength of these interactions, while effect sizes (f^2) measured their impact.
  Additionally, simple slope analysis was performed to investigate how the relationships varied at different levels of the moderators. By analyzing effects at low, medium, and high levels of moderator variables, the analysis offered detailed insights into how these factors influenced the strength and direction of the relationships.

5) Structural Model Assessment: The structural model underwent several tests to evaluate the relationships between variables. Path coefficients were analyzed for strength and significance, revealing key influences. The R^2 value indicated moderate explanatory power, while Q^2 values confirmed the model's predictive capability. Additionally, effect size (f^2) assessments highlighted the relative impact of each variable.

The overall model evaluation involved validating the conceptual framework through the ISSRM framework, identifying and operationalizing key variables into measurable indicators. A structured questionnaire collected quantitative data, refined through pilot testing for clarity and reliability. PLS-SEM was used for hypothesis testing, with assessments of reliability, validity, and structural relationships ensuring robustness. A statistically significant dataset of 139 responses (60.43% response rate) supported the analysis, meeting sampling requirements. Practical feedback from pilot testing the model with industry professionals guided refinements, while cross-referencing with literature confirmed alignment with best practices in secure

TABLE II
RESULTS AND IMPACT OF HYPOTHESES

Hypothesis	Result	Impact
H1 - CBT positively impacts OSR in trading software.	Accepted	Positive
H2 - SCP positively impacts OSR in trading software.	Accepted	Positive
H3 - SF positively impacts OSR in trading software.	Rejected	-
H4 - TMR positively impacts OSR in trading software.	Rejected	-
H5 - PC moderates the relationship between CBT and OSR.	Rejected	-
H6 - PC moderates the relationship between SCP and OSR.	Rejected	-
H7 - PC moderates the relationship between SF and OSR.	Accepted	Positive
H8 - PC moderates the relationship between TMR and OSR.	Accepted	Negative
H9 - TES moderates the relationship between TMR and OSR.	Rejected	-
H10 - TES moderates the relationship between SF and OSR.	Accepted	Negative
H11 - TES moderates the relationship between SCP and OSR.	Rejected	-
H12 - TES moderates the relationship between CBT and OSR.	Rejected	-

software development for financial trading systems.

IV. RESULTS AND DISCUSSION

The adoption of DevSecOps practices is integral to enhancing the security and resilience of financial trading systems, as it seamlessly integrates the key factors discussed in this study into a cohesive model. This research explored the interplay between various factors influencing software security through primary data and a thorough literature review. The findings provide insights into how collaborative efforts, secure coding practices, security frameworks, threat modeling and risk assessment, project complexity, and team expertise contribute to building secure and resilient trading applications.

Collaboration among teams, a fundamental aspect of DevSecOps, which is mainly falls under development, plays a crucial role in improving security outcomes. Effective communication and cooperation between developers, security professionals, and operations teams enhance the overall resilience of financial trading systems. By fostering shared responsibilities, collaborative environments enable teams to identify and address vulnerabilities promptly while ensuring consistent application of security measures. This reflects the core philosophy of DevSecOps, where cross-functional integration leads to better security outcomes.

Secure coding practices are a critical component of the development phase in DevSecOps, emerged as essential for strengthening the security of trading systems. Practices such as input validation, secure authentication, error handling, and data encryption are major instrumental components in reducing vulnerabilities. The study underscores that adherence to these practices significantly enhances the resilience of software, as developers proactively mitigate risks and ensure system reliability. Such practices provide a robust foundation for addressing common cybersecurity threats in the financial sector.

Moreover, the integration of security frameworks did not demonstrate a strong positive impact on system resilience. While security frameworks serve as a foundation for provide structure and founda- tional protocols in security measures, their effectiveness depends on adaptation and consistent application tailored to the specific demands of fast-paced trading environments. This finding suggests that rigidly following frameworks without customization may fail to address the unique challenges posed by financial trading systems.

Threat modeling and risk assessment is a major strategy of ensuring security in DevSecOps, while its valuable for identifying potential risks, showed limited direct impact on security resilience in this context. The rapidly evolving nature of cybersecurity threats in financial systems often outpaces traditional threat modeling approaches. This highlights the importance of real-time insights and continuous updates to ensure such methodologies remain effective in safeguarding trading platforms.

The study also examined how project complexity and team expertise influence the effectiveness of security practices. High project complexity was found to negatively impact the benefits of threat modeling, suggesting that traditional methods may struggle in complex environments unless adapted. However, complexity positively influenced the integration of structured security frameworks, which can act as stabilizing anchors in intricate systems. Team expertise showed mixed results; while experienced teams might rely less on formal frameworks, this can sometimes lead to overconfidence or biases, emphasizing the need for balanced approaches that combine expertise with structured security measures.

V. CONCLUSION

This research explores how DevSecOps practices enhance the security and resilience of financial trading software by emphasizing collaboration, secure coding, tailored security frameworks, and threat modeling. It introduces a novel DevSecOps model specifically designed for trading applications, exploring unique cybersecurity threats and regulatory compliance requirements. Collaborative efforts between development, security, and operations teams are highlighted as critical for identifying vulnerabilities early and ensuring consistent security measures. Secure coding practices, alongside adaptable security frameworks and threat modeling, address the complex demands of trading platforms. Validation through literature and survey data underscores the model's effectiveness, offering a specialized solution distinct from generic security approaches in financial systems.

VI. LIMITATIONS AND FUTURE WORKS

While this study provides valuable insights, several limitations must be acknowledged. The reliance on purposive sampling and a modest sample size may limit the accuracy of findings to the broader financial software industry. The

exclusive use of quantitative methods, while effective for statistical analysis, may not fully capture team dynamics and organizational challenges, which qualitative approaches could explore in greater depth. Model validation based on participant sentiment or ratings may not be sufficient to fully assess the overall security of financial trading applications, the indicators are derived from a thorough literature review, forming the foundation for the development of the questionnaire. However, subjective interpretations of these indicators might arise due to varying individual experiences or biases during the questionnaire phase. This highlights the need to incorporate more robust and objective measurement techniques to ensure the reliability and validity of the assessment outcomes.

Future research should address these limitations by using larger and more diverse samples to improve generalization and incorporating qualitative methods, such as interviews or case studies, to explore team dynamics and organizational challenges. Longitudinal studies could examine the evolution and sustained impact of DevSecOps practices, while objective security measurement techniques, like automated breach simulations or vulnerability scanning, could provide more accurate assessments. Future studies could incorporate more objective methods for model validation and evaluation. Additionally, integrating AI and machine learning into the DevSecOps model could automate secure coding practices and enhance threat detection accuracy, advancing security in trading applications.

REFERENCES

[1] R. Kumar and R. Goyal, "Modeling continuous security: A conceptual model for automated devsecops using open-source software over cloud (adoc)," *Computers & Security*, vol. 97, p. 101967, 2020.

[2] Z. A. Maher, A. Shah, S. Chan-dio, H. M. Mohadis, and N. Rahim, "Challenges and limitations in secure software development adoption-a qualitative analysis in malaysian software industry prospect," *Indian Journal of Science and Technology*, vol. 13, no. 26, pp. 2601–2608, 2020.

[3] L. Singleton, R. Zhao, M. Song, and H. Siy, "Cryptotutor: Teaching secure coding practices through misuse pattern detection," in *Proceedings of the 21st Annual Conference on Information Technology Education*, pp. 403–408, 2020.

[4] J. Mitts and E. Talley, "Informed trading and cybersecurity breaches," *Harv. Bus. L. Rev.*, vol. 9, p. 1, 2019.

[5] K. Oosthoek and C. Doerr, "Cyber security threats to bitcoin exchanges: Adversary exploitation and laundering techniques," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1616–1628, 2020.

[6] O. O. Mwambe, "Syntactic and semantic extensions of malicious activity diagrams to support issrm," *International Journal of Computer Applications*, vol. 67, no. 4, pp. 33–39, 2013.

[7] J. N. Kotey, "A functioning code may not be a secure code: A preliminary study on the students' complacency with secure coding," 2023.

[8] V. Pikulin, D. Kubo, K. Nissanka, S. Bandara, M. A. Shamsiemon, A. Yasmin, A. Jayatilaka, A. Madugalla, and T. Kanij, "Towards developer-centered secure coding training," in *2023 38th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW)*, pp. 24–31, IEEE, 2023.

[9] K. Huang, S. Madnick, N. Choucri, and F. Zhang, "A systematic framework to understand transnational governance for cybersecurity risks from digital trade," *Global Policy*, vol. 12, no. 5, pp. 625–638, 2021.

[10] P. Kariuki, L. O. Ofusori, and P. R. Subramaniam, "Cybersecurity threats and vulnerabilities experienced by small-scale african migrant traders in southern africa," *Security Journal*, pp. 1–30, 2023.

[11] Q. Liu, W. Zhang, S. Ding, H. Li, and Y. Wang, "Novel secure group data exchange protocol in smart home with physical layer network coding," *Sensors*, vol. 20, no. 4, p. 1138, 2020.

[12] O. Kayode-Ajala, "Applications of cyber threat intelligence (cti) in financial institutions and challenges in its adoption," *Applied Research in Artificial Intelligence and Cloud Computing*, vol. 6, no. 8, pp. 1–21, 2023.

[13] K. M. Hogan, G. T. Olson, J. D. Mills, and P. A. Zaleski, "An analysis of cyber breaches and effects on shareholder wealth," *International Journal of the Economics of Business*, vol. 30, no. 1, pp. 51–78, 2023.

[14] G. Gagliani, "Cybersecurity, technological neutrality, and international trade law," *Journal of International Economic Law*, vol. 23, no. 3, pp. 723–745, 2020.

[15] U. Cali, M. Kuzlu, D. J. Sebastian-Cardenas, O. Elma, M. Pipattanasomporn, and R. Reddi, "Cybersecure and scalable, token-based renewable energy certificate framework using blockchain-enabled trading platform," *Electrical Engineering*, vol. 106, no. 2, pp. 1841–1852, 2024.

[16] K. Shalabi, M. Al-Fayoumi, and Q. A. Al-Haija, "Enhancing financial system resilience against cyber threats via swift customer security framework," in *2023 International Conference on Information Technology (ICIT)*, pp. 260–265, IEEE, 2023.

[17] V. Tan, C. Cheh, and B. Chen, "From application security verification standard (asvs) to regulation compliance: A case study in financial services sector," in *2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pp. 69–76, IEEE, 2021.

[18] N. Niinivirta, "Software developers' secure coding needs in the financial sector: a case study," 2023.

[19] I. Ryan, U. Roedig, and K.-J. Stol, "Measuring secure coding practice and culture: A finger pointing at the moon is not the moon," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pp. 1622–1634, IEEE, 2023.

[20] M. Hayashi, "Secure physical layer network coding versus secure network coding," *Entropy*, vol. 24, no. 1, p. 47, 2021.

[21] J. Pruemmer, T. van Steen, and B. van den Berg, "A systematic review of current cybersecurity training methods," *Computers & Security*, p. 103585, 2023.