Systematic Review of Secure Programming Approaches in Autonomous Vehicle Software Engineering

Candra Nur Ihsan1, a), Luqman Hakim2, b), Nova Agustina3, c)

Author Affiliations

1 Research Center for Electronics

National Research and Innovation Agency, Jakarta, Indonesia

2 Universitas Muhammadiyah Malang, Malang, Indonesia.

3 Universitas Teknologi Bandung, Bandung, Indonesia

Author Emails

b) Corresponding author: luqman.hakim@umm.ac.ida) candra.nur.ihsan@brin.go.id

c) nova@utb-univ.ac.id

**Abstract.** Autonomous vehicles are complex systems that integrate various technologies, including sensors, artificial intelligence, communication, and software, to operate independently. In these systems, software functions as the core control component, making its security critically important. Software vulnerabilities serve as potential entry points for cyberattacks that endanger user safety. This study aims to examine secure software development approaches through a systematic literature review of secure programming methods applied in the context of autonomous vehicles. The review employs a Population, Intervention, Comparison, Outcome, and Context (PICOC)-based approach to evaluate the effectiveness of each method used throughout the software development lifecycle. A review of 31 selected articles reveals that the software development process for autonomous vehicles generally includes security requirement identification, system design based on security principles, secure coding practices, security testing, and continuous maintenance. The most effective methods for ensuring security include the implementation of formal verification and model checking to mathematically guarantee compliance with security specifications. Furthermore, five optimization approaches were identified as essential for strengthening secure programming practices: mitigation of return-oriented programming exploits, implementation of lightweight cryptography, use of intrusion detection systems, adoption of secure boot mechanisms and trusted execution environments, and utilization of memory-safe programming languages. This study provides a solid foundation for building robust autonomous vehicle software that is resilient to security threats.

# Introduction

This document was prepared using the AIP Conference Proceedings template for Microsoft Word. It provides a simple example of a paper and offers guidelines for preparing your article. Here we introduce the paragraph styles for Level 1, Level 2, and Level 3 headings. Please note the following:

Autonomous vehicles represent a major innovation in the transportation sector, operating independently by leveraging sensors 1, Artificial Intelligence (AI) 2,3, software 4, and advanced communication systems 5–7. The integration of these diverse technologies in autonomous vehicles necessitates strict standards in the development process to avoid functional failures that could endanger passengers. One of the most vulnerable components is the software, which governs the entire autonomous system 4. The risk of cyberattacks targeting the software in autonomous vehicles can be significantly reduced from the early development stages through the implementation of secure programming technologies designed to ensure software security 8. An established approach to address the challenges of secure programming is the Secure Software Development Life Cycle (SDLC) 9. The secure SDLC consists of five key phases: planning, requirements analysis, design, implementation, and maintenance. Each SDLC phase in the context of autonomous vehicles demands specific methods, which have been identified in prior studies. Selecting the appropriate method for each stage of the SDLC provides a solid foundation for implementing secure programming 10, aiming to ensure the security and reliability of autonomous vehicle software against potential cyber threats 11,12.

This study aims to conduct a Systematic Literature Review (SLR) on the application of secure programming in autonomous vehicle software development using the Secure SDLC approach. The review adopts the Population, Intervention, Comparison, Outcome, and Context (PICOC) method 13 to identify and analyze the methods applied in each SDLC phase within the context of autonomous vehicles. In summary, the objective of this research is:

* To identify the processes that can be implemented in autonomous vehicle software development systems based on the Secure Software Development Life Cycle (SDLC).
* To identify secure programming methods applied in the development of autonomous vehicle software.
* To identify secure programming optimization techniques that enhance software security for implementation in autonomous vehicles.

To provide a structured discussion, this article is organized as follows. First section presents related surveys on secure software development in autonomous vehicles and connected systems. Second section describes the research methodology, including the formulation of research questions, search strategies, and selection criteria based on the PICOC framework. Furthermore, next section discusses the results of the systematic literature review, which are divided into three parts: the processes implemented in the Secure SDLC, the secure programming methods applied, and the optimization techniques that support software security. Finally, Section V concludes the study by summarizing key findings and highlighting directions for future research.

# literature review

To strengthen the foundation of this study, several previous surveys and related works were reviewed to provide insights into secure software development and cybersecurity in autonomous systems. These surveys not only highlight the current state of research but also present challenges and gaps that align with the objectives of this paper. The following subsections summarize three key studies that are closely related to our research focus, i.e., secure software lifecycle approaches for autonomous vehicles, cybersecurity issues in unmanned aerial vehicles, and security attacks and defense techniques in connected and autonomous vehicles.

## Towards a Secure Software Lifecycle for Autonomous Vehicles 4

Article 4 introduces Secure Vehicle Software Engineering (SVSE) as a secure software lifecycle model for autonomous vehicles. This model ensures security from the early stages of development to the operational phase by integrating security measures into every stage of the development process. SVSE also complies with international standards such as ISO/SAE 21434 and WP.29, ensuring that autonomous vehicles remain resilient against evolving cyber threats throughout their operational lifespan. Article 4 aligns with our research in its focus on the secure software development lifecycle for autonomous vehicles. However, while it emphasizes the application of security standards across the entire SDLC, our study aims to conduct a systematic review of various secure programming approaches implemented in each phase of the SDLC. This enables a deeper evaluation of specific secure coding methods that can be applied to strengthen software security in autonomous vehicle systems.

## Cybersecurity of Unmanned Aerial Vehicles: A Survey 11

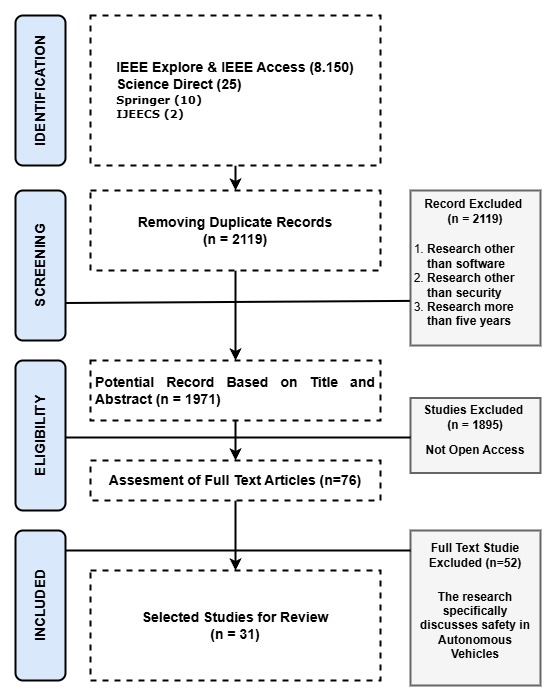
Article 11 reviews cybersecurity issues in Unmanned Aerial Vehicles (UAVs), which are increasingly used across various sectors. UAVs face security challenges similar to those of autonomous vehicles, including attacks targeting communication systems and software. The article discusses potential threats and mitigation techniques such as software vulnerability analysis and GPS spoofing attacks, which can also be applied to other autonomous systems, including autonomous vehicles. Although Article 11 focuses on UAVs, many of the security challenges are similar to those encountered by autonomous vehicles, particularly regarding communication and control systems that are vulnerable to attacks. Our research, however, is specifically focused on autonomous ground vehicles and the application of secure programming methods in software development. While our study systematically investigates secure coding techniques within the Secure SDLC framework to address cybersecurity threats, this article places greater emphasis on cybersecurity in the context of hardware and communication systems.

## A Survey on Security Attacks and Defense Techniques for Connected and Autonomous Vehicles 14

Article 14 provides a comprehensive review of security attacks and defense techniques in autonomous and connected vehicles (CAVs). The study classifies attack models based on the targeted vehicle components and discusses defense strategies such as encryption and anomaly detection to enhance CAV system security. The article highlights emerging security challenges, including attacks on sensors and communication networks, which must be addressed by autonomous vehicle developers. While both Article 14 and our study focus on autonomous vehicles, they adopt different approaches. Article [9] primarily discusses general defense tactics and physical vulnerabilities in components such as Electronic Control Units (ECUs) and communication networks. In contrast, our research concentrates on a systematic review of secure coding methods implemented throughout the Secure SDLC to ensure software security. We examine the specific techniques applicable to each phase of software development, whereas Article [9] provides a broader overview of attacks and countermeasures.

# Methods

The methodology for this systematic literature review on secure programming approaches in autonomous vehicle software engineering adopts the Population, Intervention, Comparison, Outcome, and Context (PICOC) framework to ensure a comprehensive and unbiased assessment of existing research. This chapter outlines the systematic search strategy employed, including the selection criteria for relevant studies and the keywords used. It also describes the data extraction process, the techniques applied to evaluate the quality and relevance of each study, and the methods used to synthesize the extracted findings. The results of the literature search are summarized in Figure 1.



**Figure 1.** PICOC Flow for Study Selection in the Systematic Literature Review on Secure Programming in Autonomous Vehicles

## Research Question

In this study, research questions were formulated to guide the systematic literature review process and to support the identification, analysis, and synthesis of findings related to secure programming approaches in autonomous vehicle software engineering. The Population, Intervention, Comparison, Outcome, and Context (PICOC) framework was applied to ensure that every aspect of the research is analyzed in a systematic and structured manner. Table 1 presents the formulation of the research questions developed using the PICOC framework.

|  |  |
| --- | --- |
| **TABLE 1.** Research Question Formulation Using The Picoc Framework | |
| **PICOC Attributes** | **Description** |
| Population | Autonomous vehicle software development |
| Intervention | Implementation of secure programming in the SDLC |
| Comparison | Comparison of secure programming methods |
| Outcomes | Effectiveness in reducing software vulnerabilities |
| Context | Autonomous vehicles and security standards |

Subsequently, the research questions were formulated with a focus on critical aspects of secure software development, particularly in the context of cybersecurity and the challenges faced by developers in ensuring the security of autonomous vehicle software. The formulated research questions are as follows:

* What processes are implemented in autonomous vehicle software development systems based on the Secure Software Development Life Cycle (SDLC)?
* What secure programming methods are applied in the development of autonomous vehicle software?
* What secure programming optimizations can help enhance software security?

## Search Strategies

The literature search was conducted across several major scientific databases, including IEEE Xplore and IEEE Access, ScienceDirect, Springer, and JEECS. This study involved data extraction, quality assessment of articles based on their relevance and contribution to the topic of secure programming, and data synthesis using a descriptive qualitative approach to identify trends, research gaps, and opportunities for further investigation in this field.

## Selection Criteria and Process

The selection process in this study followed a systematic approach to ensure that only the most relevant and high-quality studies were included in the review. The selection criteria were defined based on the PICOC (Population, Intervention, Comparison, Outcome, and Context) framework, aiming to filter the literature objectively and in a structured manner. Studies were included in the review if they met the criteria summarized in Table 2.

|  |  |  |
| --- | --- | --- |
| **TABLE 2.** Inclusion And Exclusion Criteria for Study Selection | | |
| **Category** | **Inclusion Criteria** | **Exclusion Criteria** |
| Topic | Studies discussing software for autonomous vehicles or related cyber-physical systems | Studies focusing solely on hardware without any software-related aspects |
| Accessibility | Articles available through open access | Articles not openly accessible (restricted or paid access) |
| Publication Date | Published within the last five years | Published more than five years ago |
| Language | Articles written in English | Articles written in languages other than English |

The literature search was conducted across several major scientific databases, yielding 8,150 articles from IEEE Xplore and IEEE Access, 25 from ScienceDirect, 10 from Springer, and 2 from JEECS, resulting in a total of 8,187 initial articles. After the removal of duplicates (n = 2,119), a screening process was performed based on titles and abstracts. Articles that were not relevant to the topic, such as those unrelated to software, security, or published more than five years ago, were excluded (n = 2,119). The next selection stage involved accessibility, considering only open access articles. A total of 1,895 articles were excluded due to restricted access. Of the 76 articles that advanced to full-text evaluation, 52 were eliminated for not explicitly addressing safety aspects in the context of autonomous vehicles. As a result, 31 articles were selected for in-depth review.

# Result and Discussion

This section presents the discussion of 31 articles relevant to the development of autonomous vehicle software based on secure programming. The analysis was conducted to address the three established research questions (RQs), namely the processes involved in the Secure SDLC, the most effective methods applied, and the optimization techniques that support software security.

## What processes are implemented in autonomous vehicle software development systems based on the Secure Software Development Life Cycle (SDLC)?

Based on the literature review, most studies adopt the Secure Software Development Life Cycle (Secure SDLC) approach, which consists of several phases summarized in Table 3.

|  |  |  |
| --- | --- | --- |
| **TABLE 3.** Secure SDLC Processes | | |
| **Category** | **Description** | **Reference** |
| Requirement & Threat Modeling | Identification of security requirements and threat modeling. | 15–18 |
| Secure Design | System design incorporates the principles of least privilege, defense in depth, and fail-safe defaults. | 19 |
| Secure Coding | Coding techniques, formal verification, model checking, and static code analysis help reduce exploit risks. | 20–23 |
| Security Testing | The testing phase involves approaches such as penetration testing, fuzz testing, and dynamic analysis. | 9,24,25 |
| Maintenance & Monitoring | Security maintenance and post-deployment monitoring after the software is implemented. | 26 |

The review findings indicate that the Secure SDLC approach in autonomous vehicle software development is not only based on standard frameworks such as NIST or OWASP, but also tailored to the specific needs of autonomous systems, which are real-time and safety-critical in nature. A fully implemented Secure SDLC can reduce system vulnerabilities by up to 60%, according to experimental results in study 27. Another key finding is that post-deployment security monitoring remains largely overlooked in the automotive industry, making it a critical area for further research.

## What secure programming methods are applied in the development of autonomous vehicle software?

The methods identified were assessed based on their ability to detect vulnerabilities at early development stages 27–32,or through vulnerability detection in source code 33–35. To ensure the functional safety of complex and critical systems such as autonomous vehicles, formal verification is a commonly adopted method 36,37. Formal verification applies mathematical techniques such as model checking to validate that the system meets strict security specifications, including aspects of real-time control and interactions among subsystems. Recent studies show that Bounded Model Checking (BMC) applied to hierarchical Synchronous-Reactive (SR) models like Ptolemy II provides efficient verification results 38, capable of detecting bugs or logic failures before the system is deployed in real-world environments. Furthermore, formal specification methods such as CafeOBJ and Maude support specification translation, enabling the combined use of theorem proving and model checking 37. These findings offer both flexibility and strong verification capabilities in developing autonomous systems that exhibit a combination of discrete and continuous behaviors. Through the integration of these methods, autonomous vehicle systems can be built with higher levels of security assurance, accuracy, and reliability.

## What secure programming optimizations can help enhance software security?

The results of this literature review identified five optimization methods that can be effectively applied in secure programming practices, particularly in the development of autonomous vehicle software. The distribution of these methods is illustrated in Figure 2.

**Figure 2.** Distribution of Security Approaches in Automotive Cyber-Physical Systems

In Figure 2, Return-Oriented Programming (ROP) 15 mitigation is used to prevent exploit attacks that reuse memory instructions, through techniques such as Address Space Layout Randomization (ASLR) and runtime execution validation 29. Next, Lightweight Cryptography offers efficient and resource-saving encryption solutions 25,31,39,40, making it highly suitable for embedded devices and real-time systems such as Electronic Control Units (ECUs) in vehicles. Another method involves the implementation of Artificial Intelligence-based Intrusion Detection Systems (IDS), which provide early detection of suspicious activities and anomaly attacks 41–43 that could compromise the system, including vehicle-to-everything (V2X) communication 17. Additionally, Secure Boot and Trusted Execution Environments (TEE) ensure software integrity from the initial boot process and provide an isolated execution environment for running critical functions securely 44,45. Lastly, the use of Memory-Safe Programming Languages, such as Rust, replaces languages prone to memory errors, thereby reducing the likelihood of security vulnerabilities caused by buffer overflows and invalid pointers.

# Conclusion

This literature review reveals that secure software development in autonomous vehicle systems heavily relies on the implementation of systematic and structured processes. These processes include security requirements and threat analysis, system design based on the principles of caution and redundancy, security-aware coding practices, as well as continuous testing and monitoring. In the safety-critical context of autonomous vehicles, mathematical approaches such as formal verification and model checking have proven to be effective in ensuring that software meets stringent security specifications. Additionally, five secure programming optimization techniques were identified as significantly contributing to enhanced software security: mitigation of return-oriented programming (ROP) exploits, implementation of lightweight and efficient cryptography, deployment of artificial intelligence-based intrusion detection systems, application of secure boot mechanisms and trusted execution environments, and the use of memory-safe programming languages designed to prevent memory management errors. These findings underscore the importance of a comprehensive and layered approach to designing and implementing autonomous vehicle software capable of addressing complex and evolving security challenges.

Future research is recommended to explore the practical implementation and empirical evaluation of the identified secure programming methods, particularly within simulation environments or prototype autonomous vehicle systems. Furthermore, the development of frameworks that automatically integrate formal verification into the software development pipeline of autonomous vehicles presents a promising research opportunity. Subsequent studies may also expand the scope by considering security integration in vehicle-to-vehicle communication components and examining the impact of interoperability on overall system vulnerabilities.

# Acknowledgments (Use the Microsoft Word template style: *Heading 1*) or (Use Times New Roman Font: 12 pt, Bold, ALL CAPS, Centered)

If desired, in this section we welcome you to include thank your comments for those who have supported your research. Font should be Times New Roman, 10 pt.

# References

1. 1 Y. Cao, Y. Zhou, Q.A. Chen, C. Xiao, W. Park, K. Fu, B. Cyr, S. Rampazzi, and Z. Morley Mao, “Adversarial sensor attack on LiDAR-based perception in autonomous driving,” in Proceedings of the ACM Conference on Computer and Communications Security, (Association for Computing Machinery, 2019), pp. 2267–2281.
2. 2 D. Jivani, H. Gajjar, R. Gupta, N.K. Jadav, S. Tanwar, J. Bhatia, and H. Trivedi, “Machine Learning-based Secure Communication Framework for Connected Autonomous Vehicles,” in 2023 IEEE Globecom Workshops, GC Wkshps 2023, (Institute of Electrical and Electronics Engineers Inc., 2023), pp. 359–364.
3. 3 R. Gundu, and M. Maleki, “Securing CAN Bus in Connected and Autonomous Vehicles Using Supervised Machine Learning Approaches,” in IEEE International Conference on Electro Information Technology, (IEEE Computer Society, 2022), pp. 42–46.
4. 4 L.J. Moukahal, M. Zulkernine, and M. Soukup, “Towards a Secure Software Lifecycle for Autonomous Vehicles,” in Proceedings - 2021 IEEE International Symposium on Software Reliability Engineering Workshops, ISSREW 2021, (Institute of Electrical and Electronics Engineers Inc., 2021), pp. 371–377.
5. 5 A. Sersemis, A. Papadopoulos, G. Spanos, A. Lalas, K. Votis, and D. Tzovaras, “Cybersecurity Oriented Architecture to Ensure the Autonomous Vehicles Communication,” in 2023 31st Mediterranean Conference on Control and Automation, MED 2023, (Institute of Electrical and Electronics Engineers Inc., 2023), pp. 119–124.
6. 6 J.F. Rusdi, M. Naseer, B. Pudjoatmodjo, S.A. Sidik, I.N. Aziz, and R. Hardi, “Motor Cycle SmartKey System,” in 2021 3rd International Conference on Cybernetics and Intelligent System (ICORIS), (IEEE, 2021), pp. 1–5.
7. 7 R. Hardi, N. Suryana, N.C. Pee, A.S. Pribadi, J.F. Rusdi, and A. Junaidi, “The Object Tracking System at the Service Delivery Center of the Traveling Salesperson Problem Method,” in J Phys Conf Ser, (IOP Publishing Ltd, 2021).
8. 8 M. Naseer, J.F. Rusdi, N.M. Shanono, S. Salam, Z. Bin Muslim, N.A. Abu, and I. Abadi, “Malware Detection: Issues and Challenges,” in J Phys Conf Ser, (IOP Publishing Ltd, 2021).
9. 9 A.M. Fernandes, A. Pai, and L.M.M. Colaco, “Secure SDLC for IoT Based Health Monitor,” in 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), (IEEE, 2018), pp. 1236–1241.
10. 10 A. Ramirez, A. Aiello, and S.J. Lincke, “A survey and comparison of secure software development standards,” in 13th CMI Conference on Cybersecurity and Privacy - Digital Transformation - Potentials and Challenges, CMI 2020, (Institute of Electrical and Electronics Engineers Inc., 2020).
11. 11 R. Guo, M. Huang, J. Li, and J. Wang, “Cybersecurity of Unmanned Aerial Vehicles: A Survey,” in Proceedings - 2021 14th International Conference on Advanced Computer Theory and Engineering, ICACTE 2021, (Institute of Electrical and Electronics Engineers Inc., 2021), pp. 57–64.
12. 12 J.F. Rusdi, S. Salam, N.A. Abu, T.G. Baktina, R. Gumilar Hadiningrat, B. Sunaryo, A. Rusmartiana, W. Nashihuddin, P. Fannya, F. Laurenty, N.M. Shanono, R. Hardi, S. Kuswayati, S.E. Damayanti, and S. Rahmawati, “Collaborative of ICT Research in Indonesia,” in J Phys Conf Ser, (IOP Publishing Ltd, 2021).
13. 13 N. Agustina, Kusrini, E. Utami, and T. Hidayat, “Systematic Literature Review in the Development of Datasets and Fact Verification Models for Indonesian Language,” in 2024 7th International Conference of Computer and Informatics Engineering (IC2IE), (IEEE, 2024), pp. 1–9.
14. 14 M. Pham, and K. Xiong, “A Survey on Security Attacks and Defense Techniques for Connected and Autonomous Vehicles,” (2020).
15. 15 R.K. Shrivastava, S.P. Singh, M.K. Hasan, Gagandeep, S. Islam, S. Abdullah, and A.H.M. Aman, “Securing Internet of Things devices against code tampering attacks using Return Oriented Programming,” Comput Commun 193, 38–46 (2022).
16. 16 A.K. Talukder, V.K. Maurya, B.G. Santhosh, E. Jangam, S. V. Muni, K.P. Jevitha, S. Saurabh, and A.R. Pais, “Security-aware Software Development Life Cycle (SaSDLC) - Processes and tools,” in 2009 IFIP International Conference on Wireless and Optical Communications Networks, (IEEE, 2009), pp. 1–5.
17. 17 S. Jain, N.J. Ahuja, P. Srikanth, K.V. Bhadane, B. Nagaiah, A. Kumar, and C. Konstantinou, “Blockchain and Autonomous Vehicles: Recent Advances and Future Directions,” IEEE Access 9, 130264–130328 (2021).
18. 18 R.A. Khan, S.U. Khan, H.U. Khan, and M. Ilyas, “Systematic Mapping Study on Security Approaches in Secure Software Engineering,” IEEE Access 9, 19139–19160 (2021).
19. 19 O.O. Olusanya, R.G. Jimoh, S. Misra, and J.B. Awotunde, “A neuro-fuzzy security risk assessment system for software development life cycle,” Heliyon 10(13), (2024).
20. 20 K. Grosse, and A. Alahi, “A qualitative AI security risk assessment of autonomous vehicles,” Transp Res Part C Emerg Technol 169, (2024).
21. 21 R. Kwon, and G. Kwon, “Formal Verification of STPA with Model Checking,” Scientific Journal of Gdynia Maritime University (125), 7–24 (2023).
22. 22 X. Yang, X. Chen, and J. Wang, “A Model Checking Based Software Requirements Specification Approach for Embedded Systems,” in Proceedings - 31st IEEE International Requirements Engineering Conference Workshops, REW 2023, (Institute of Electrical and Electronics Engineers Inc., 2023), pp. 184–191.
23. 23 S. Wang, M. Yang, S. Jiang, F. Chen, Y. Zhang, and X. Fu, “BBS: A secure and autonomous blockchain-based big-data sharing system,” Journal of Systems Architecture 150, (2024).
24. 24 K. Ren, Q. Wang, C. Wang, Z. Qin, and X. Lin, “The Security of Autonomous Driving: Threats, Defenses, and Future Directions,” Proceedings of the IEEE 108(2), 357–372 (2020).
25. 25 D.K.P. Newar, R. Zhao, H. Siy, L.K. Soh, and M. Song, “SSDTutor: A feedback-driven intelligent tutoring system for secure software development,” Sci Comput Program 227, (2023).
26. 26 A. Kooshari, and M. Fartash, “A Distributed and Secure Software Architecture Based on Blockchain Technology for Application Software,” Wirel Pers Commun 130(1), 219–240 (2023).
27. 27 W. Charoenwet, P. Thongtanunam, V.T. Pham, and C. Treude, “Toward effective secure code reviews: an empirical study of security-related coding weaknesses,” Empir Softw Eng 29(4), (2024).
28. 28 R.A. Bakar, F. Paolucci, F. Cugini, P. Castoldi, and J. Jose Vegas Olmos, “DPUAUT: Secure Authentication Protocol with SmartNiC Integration for Trustworthy Communications in Intelligent Swarm Systems,” IEEE Access 12, 89986–90004 (2024).
29. 29 S.-J. Chen, Y.-C. Pan, Y.-W. Ma, and C.-M. Chiang, “The Impact of the Practical Security Test during the Software Development Lifecycle,” in 2022 24th International Conference on Advanced Communication Technology (ICACT), (IEEE, 2022), pp. 313–316.
30. 30 Q. Rouland, B. Hamid, and J. Jaskolka, “A model-driven formal methods approach to software architectural security vulnerabilities specification and verification,” Journal of Systems and Software 219, (2025).
31. 31 J. Cheng, D. Xu, G. Srivastava, and K. Yu, “Packet Encoding Based on Encrypted Raptor Code for Secure Internet of Vehicles Communication,” in IEEE Vehicular Technology Conference, (Institute of Electrical and Electronics Engineers Inc., 2023).
32. 32 D.K. Davis, and F. Zhu, “Eye tracking technologies to visualize secure coding behavior,” Array 15, (2022).
33. 33 S.A. Ebad, “Exploring How to Apply Secure Software Design Principles,” IEEE Access 10, 128983–128993 (2022).
34. 34 Z. Fang, H. Fu, T. Gu, Z. Qian, T. Jaeger, P. Hu, and P. Mohapatra, “A model checking-based security analysis framework for IoT systems,” High-Confidence Computing 1(1), (2021).
35. 35 X. Fang, R. Calinescu, S. Gerasimou, and F. Alhwikem, “Fast Parametric Model Checking with Applications to Software Performability Analysis,” IEEE Transactions on Software Engineering 49(10), 4707–4730 (2023).
36. 36 M.S. Saeed, and S. Bezzateev, “A Formal Verification-Based Risk Assessment Framework for the Authentication of Self-driving Vehicles,” in Proceedings of the 2024 8th International Conference on Information, Control, and Communication Technologies, ICCT 2024, (Institute of Electrical and Electronics Engineers Inc., 2024).
37. 37 Y. Wang, M. Nakamura, and K. Sakakibara, “Investigation of Formal Verification of the Autonomous Vehicle Control System by Specification Translation,” in 2023 International Technical Conference on Circuits/Systems, Computers, and Communications, ITC-CSCC 2023, (Institute of Electrical and Electronics Engineers Inc., 2023).
38. 38 X. Zhang, Z. Yang, H. Kong, and W. Kong, “Formal Verification of Hierarchical Ptolemy II Synchronous-Reactive Models with Bounded Model Checking,” in IEEE International Conference on Software Quality, Reliability and Security, QRS, (Institute of Electrical and Electronics Engineers Inc., 2022), pp. 410–421.
39. 39 S. Prokić, N. Luburić, J. Slivka, and A. Kovačević, “Prescriptive procedure for manual code smell annotation,” Sci Comput Program 238, (2024).
40. 40 T. Islam, Md.A. Sheakh, A.N. Jui, O. Sharif, and M.Z. Hasan, “A review of cyber attacks on sensors and perception systems in autonomous vehicle,” Journal of Economy and Technology 1, 242–258 (2023).
41. 41 L. Babun, K. Denney, Z.B. Celik, P. McDaniel, and A.S. Uluagac, “A survey on IoT platforms: Communication, security, and privacy perspectives,” Computer Networks 192, (2021).
42. 42 P. Nath, J.R. Mushahary, U. Roy, M. Brahma, and P.K. Singh, “AI and Blockchain-based source code vulnerability detection and prevention system for multiparty software development,” Computers and Electrical Engineering 106, (2023).
43. 43 Y. Valdés-Rodríguez, J. Hochstetter-Diez, M. Diéguez-Rebolledo, A. Bustamante-Mora, and R. Cadena-Martínez, “Analysis of Strategies for the Integration of Security Practices in Agile Software Development: A Sustainable SME Approach,” IEEE Access 12, 35204–35230 (2024).
44. 44 V. Casola, A. De Benedictis, C. Mazzocca, and V. Orbinato, “Secure software development and testing: A model-based methodology,” Comput Secur 137, (2024).
45. 45 H. Nina, J.A. Pow-Sang, and M. Villavicencio, “Systematic Mapping of the Literature on Secure Software Development,” IEEE Access 9, 36852–36867 (2021).