Towards Autonomous Detection and Hot Patching of Binary Vulnerabilities in UAV Software

Iryna Stopochkina, Mykola Ilin, Oleksii Novikov, Andrii Voitsekhovskyi
Igor Sikorsky Kyiv Polytechnic Institute
Kyiv, Ukraine
i.stopochkina@kpi.ua

Summary

Unmanned aerial vehicles increasingly rely on embedded firmware, autopilot components, communication modules, telemetry-processing libraries, and ground-station software. Vulnerabilities in these components may affect UAV control, telemetry integrity, mission execution, and resilience against malicious interference [1]. In many practical scenarios, including incident response, forensic analysis, and security assessment of deployed UAV systems, researchers and analysts do not have access to the original source code. Instead, they must work with binary firmware images, executable files, libraries, or extracted software samples.

Recent studies have shown that large language models can support vulnerability detection, malware reverse engineering, code comprehension, and software security analysis tasks [2–4]. The vulnerability-detection stage for UAV-oriented software was addressed in our previous work [5]. Therefore, the present study focuses on the next stage of the security workflow: generation and validation of candidate binary-level patches for vulnerable UAV software components.

The main contribution of this work is a validation-oriented binary hot-patching pipeline for UAV software, where LLM-assisted reverse engineering is combined with static analysis, dynamic instrumentation, exploit reproduction, and analyst-controlled approval. The proposed approach starts from a previously identified vulnerability and generates a candidate patch as a minimal modification of the affected binary component. The patch is not treated as automatically correct. Instead, the validation stage attempts to reproduce the original vulnerability or an equivalent exploitable condition in the patched binary.

For this purpose, the pipeline combines decompiled-code analysis, call-graph inspection, system-resource analysis, dynamic instrumentation, execution tracing, and LLM-assisted hypothesis refinement. The LLM agent supports the analyst by formulating test scenarios, generating instrumentation scripts, interpreting runtime errors, and refining the validation strategy. If the exploitable condition cannot be reproduced under predefined static and dynamic tests, the patch is submitted for expert review as an approved candidate fix.

The proposed methodology is intended to support rapid preparation and validation of security fixes for UAV software under source-code unavailability. In this paper, “hot patching” is understood primarily as rapid binary-level remediation, not as unrestricted in-flight firmware replacement. Final deployment must still consider flight safety, real-time constraints, state consistency, communication reliability, and certification requirements. The proposed approach does not replace expert review, formal verification, certification, or flight testing. However, it can reduce the time required to move from vulnerability detection to a validated candidate fix.

The preliminary evaluation demonstrates the feasibility of the proposed pipeline on open UAV software components, including PX4 and ArduPilot-related binaries [6], [7]. The obtained results indicate that LLM-assisted binary analysis, when combined with controlled dynamic validation and expert approval, can support practical remediation workflows for security-critical embedded UAV systems.

References

[1] O. Novikov, I. Stopochkina, A. Voitsekhovskyi, M. Ilin, and M. Ovcharuk, “Simulation of UAV networks on the battlefield, taking into account cyber-physical influences that affect availability,” *Theoretical and Applied Cyber Security*, vol. 6, no. 2, pp. 66–76, 2024. doi: https://doi.org/10.20535/tacs.2664-29132024.2.318182

[2] X. Zhou, T. Zhang, and D. Lo, “Large Language Model for Vulnerability Detection: Emerging Results and Future Directions,” arXiv 2401.15468, 2024. [Online]. Available: https://arxiv.org/abs/2401.15468/

[3] A. Williamson and M. Beauparlant, “Malware Reverse Engineering with Large Language Model for Superior Code Comprehensibility and IoC Recommendations,” 2024. Preprint. doi: https://doi.org/10.21203/rs.3.rs-4471373/v1

[4] Y. Li et al., “Everything You Wanted to Know About LLM-based Vulnerability Detection But Were Afraid to Ask,” arXiv2504.13474, 2025. [Online]. Available: https://arxiv.org/abs/2504.13474

[5] Voiteskhovskyi A., Stopochkina I., Sun P., Xie J., Ilin M., & Novikov O. (2026). Detection of vulnerabilities in software for unmanned aerial vehicles by using large language models. *Eastern-European Journal of Enterprise Technologies*, vol. 1, no. 2(139), pp. 36–47, 2026. https://doi.org/10.15587/1729-4061.2026.352029

[6] Dronecode Foundation, “PX4 Autopilot.” [Online]. Available: https://px4.io/

[7] ArduPilot Project, “What is ArduPilot?” [Online]. Available: https://ardupilot.org/