DLL Hijacking and Defensive Measures in Windows Applications

Hoàng Văn Hùng
Thai Nguyen University of Information and Communication Technology
Thai Nguyen University, Thai Nguyen, Vietnam
student626@ictu.edu.vn

***Abstract*—This undergraduate thesis investigates DLL hijacking techniques and defensive measures in Windows applications. Windows applications can load shared libraries through search paths that require careful configuration and monitoring. The student surveys dynamic-link libraries, search order, sideloading and injection risks, then documents controlled demonstrations and defensive practices in a Windows test environment. The manuscript presents an example of DLL injection into a local application and screenshots of file effects, alongside a discussion of safer loading, access limits, and system protection. The conclusion discusses game cheating and malicious injection more generally than the cover topic. The source does not show a systematic defense evaluation or production incident data; its demonstrations must be read as controlled examples. The contribution is a documented student study or prototype whose stated scope can guide further testing.**

***Index Terms*—DLL hijacking, Windows security, DLL injection, defense**

I. Introduction

Windows applications can load shared libraries through search paths that require careful configuration and monitoring. This thesis investigates DLL hijacking techniques and defensive measures in Windows applications. It begins from a practical engineering need and explains the technologies and workflow chosen for the case. The objective is to document a student implementation or comparative investigation, together with the evidence available in the manuscript. Broader operational or economic effects require separate measurements and are not assumed from the existence of a prototype [1]. Earlier ICTU student work on related designs is included here for context [2], [3], [4]. Related published studies provide context for the subject and methods [5], [6], [7], [8], [9], [10]. These contextual works were selected for this English summary; they are not asserted to appear in the original thesis bibliography.

II. Methods

The student surveys dynamic-link libraries, search order, sideloading and injection risks, then documents controlled demonstrations and defensive practices in a Windows test environment. The source is summarized through its stated design, implementation material, examples, and conclusion. Features shown in the manuscript are distinguished from proposals for later development. This review does not add a new experiment, fill missing measurements with estimates, or treat a planned capability as an observed result. The source thesis remains the provenance for the reported work [1].

Supervisor: Vũ Việt Dũng, Thai Nguyen University of Information and Communication Technology (vvdung@ictu.edu.vn).

III. Results

The manuscript presents an example of DLL injection into a local application and screenshots of file effects, alongside a discussion of safer loading, access limits, and system protection. The report's screens, configurations, examples, or measurements constitute the evidence for this summary. A capability described only as a design intention is not counted as a validated operational outcome. No additional performance, adoption, financial, or safety figure has been introduced here. The result is interpreted within the scope of the student's documented setup [1].

IV. Discussion

The conclusion discusses game cheating and malicious injection more generally than the cover topic. The source does not show a systematic defense evaluation or production incident data; its demonstrations must be read as controlled examples. Further work could compare mitigations under a documented test matrix and measure whether the recommended controls block each demonstrated loading path. These limitations affect how the evidence should be read. The current manuscript can support understanding of the design and reported demonstration, while independent tests would be needed for stronger claims. Any later deployment should document its environment and separate observed outcomes from intended benefits [1].

V. Conclusion

The thesis documents DLL hijacking techniques and defensive measures in Windows applications. The manuscript presents an example of DLL injection into a local application and screenshots of file effects, alongside a discussion of safer loading, access limits, and system protection. The conclusion discusses game cheating and malicious injection more generally than the cover topic. The source does not show a systematic defense evaluation or production incident data; its demonstrations must be read as controlled examples. Further work could compare mitigations under a documented test matrix and measure whether the recommended controls block each demonstrated loading path. Accordingly, the present

contribution is a source-grounded account of the student work and its tested or illustrated scope. Claims about broader effectiveness should be assessed in later work using explicit evaluation conditions [1].

References

[1] Hoàng Văn Hùng, ”DLL Hijacking and Defensive Measures in Windows Applications,” internal undergraduate thesis (original in Vietnamese), Thai Nguyen University of Information and Communication Technology, 2024.

[2] ICTU Student 129, ”Construction of a Network Security System for the Commercial Bank of Vietnam Hanoi Branch,” internal undergraduate thesis, Thai Nguyen University of Information and Communication Technology, 2009.

[3] ICTU Student 130, ”Construction of a Network Security System for a College of Education in Thai Nguyen,” internal undergraduate thesis, Thai Nguyen University of Information and Communication Technology, 2009.

[4] ICTU Student 151, ”Information Security in E-commerce,” internal undergraduate thesis, Thai Nguyen University of Information and Communication Technology, 2010.

[5] JunSeok Koo et al., ”A research on detection techniques of Proxy DLL malware disguised as a Windows library : Focus on the case of Winnti,” Journal of the Korea Institute of Information Security and Cryptology, 2015. doi:10.13089/jkiisc.2015.25.6.1385.

[6] Mohammad Afiq Amirul Abdul Rauf et al., ”Malware Behaviour Analysis and Classification via Windows DLL and System Call,” Journal of Physics: Conference Series, 2020. doi:10.1088/1742-6596/1529/2/022097.

[7] Pedro Fernández-Álvarez et al., ”Module extraction and DLL hijacking detection via single or multiple memory dumps,” Forensic Science International: Digital Investigation, 2023. doi:10.1016/j.fsidi.2023.301505.

[8] J. Berdajs et al., ”Extending applications using an advanced approach to DLL injection and API hooking,” Software: Practice and Experience, 2010. doi:10.1002/spe.973.

[9] Le An et al., ”An empirical study of DLL injection bugs in the Firefox ecosystem,” Empirical Software Engineering, 2019. doi:10.1007/s10664-018-9677-7.

[10] Digvijaysinh Rathod et al., ”Defense against Crypto-ransomwares families using Dynamic Binary Instrumentation and DLL injection,” International Journal of Electronic Security and Digital Forensics, 2023. doi:10.1504/ijesdf.2023.10052498.