Peer Response 2

Hi Andrius,

Thank you for your submission to the discussion, and exploring the Injection weakness defined in OWASP Top 10. Surprisingly, it still remains a relatively common weakness in many applications (OWASP, 2021) and can have grave implications to data security.

The use of sequence diagram may not be the most sufficient to highlight the range of potential weaknesses that may be introduced during a decision-making process. Sequence diagrams are useful in design and testing scenarios when demonstrating how software communicates when performing a particular task, but it can be limiting due to the vertical-only nature of the diagram (Al-Fedaghi, 2021). I believe this may have hindered assessing a wider range of potential interactions between the user, controller, database and its interface.

With respect to the messages currently in the sequence diagram, it is critical to demonstrate how injected code can be validated and then sanitised, as some input validators are more sufficient than others. For example, Ray and Ligatti (2012) demonstrate how it is not necessarily "code" as we understand it which may cause injection attacks, therefore existing solutions must consider the potential consequences of all untrusted inputs.

An interesting example of an application susceptible to injection attacks would be one that uses Large Language Models (LLMs) to implement a chat-bot feature for example. LLMs have been demonstrated to be vulnerable to prompt injection attacks (Liu et al.,

2023) and given the proliferation of applications using GPT models, this can cause a new wave of security issues.

Thank you again for your contribution to this discussion. It has been valuable to revisit the concept of injection attack and assess it in the wider context of secure software development. Additionally, it raises the discussion on sufficient validation and sanitisation methods, especially given how effortlessly large bulks of text can be generated, hiding potentially malicious commands in a range of forms.

References:

Al-Fedaghi, S. (2021) UML sequence diagram: an alternative model. *arXiv preprint arXiv:2105.15152*. Available from: https://arxiv.org/pdf/2105.15152 [Accessed 4 November 2024]

Liu, Y., Deng, G., Li, Y., Wang, K., Wang, Z., Wang, X., Zhang, T., Liu, Y., Wang, H., Zheng, Y. & Liu, Y. (2023) Prompt Injection attack against LLM-integrated Applications. *arXiv preprint arXiv:2306.05499*. Available from: https://arxiv.org/pdf/2306.05499 [Accessed 4 November 2024]

OWASP (2021) A03:2021 – Injection. OWASP Top 10. Available from: https://owasp.org/Top10/A03_2021-Injection/ [Accessed 4 November 2024]

Ray, D. & Ligatti, J. (2012) Defining code-injection attacks. *Acm Sigplan Notices*, *47*(1): 179-190.