ABSTRACT

Graph Query Language (GraphQL) is a query language that determines how clients interact with an Application Programming Interface (API). The main purpose of creating GraphQL is to facilitate data communication between the backend and frontend, where GraphQL can provide complete and easy to understand data descriptions. Over time, GraphQL's popularity continues to increase due to its powerful capabilities in managing and retrieving API data. Like other technologies, GraphQL also has several weak points which can cause vulnerabilities, one of which is injection vulnerabilities. This research was carried out with the aim of finding which injection technique is most effective with two security modes, namely hardening and before hardening by comparing which injection technique takes less time so that in the future we can find the most effective security solution. In this exploitation research, three different injection techniques were used, namely Command Injection, Log Injection and Spoof Injection. Exploitation of GraphQL using injection techniques is formulated in the form of an attack tree diagram with the aim of knowing the exploit tree relationship based on time metrics. From the results of the exploitation time of the three injection techniques, it was found that spoof injection required the shortest time in security mode before hardening with a total time of 32.63s. And the injection technique that takes quite a long time is command injection with a total time of 60.15s. In security after hardening, there was a change in time where log injection was in the first fastest place with a total time of 38.19s and in the last place the injection technique which took quite a long time was the command injection injection technique with a total time of 53.52s. So it can be concluded that the injection attack that is effective before hardening is spoof injection, while the injection attack that is effective after hardening is log injection, where from this injection attack in the future the most effective security solution can be found.

Keywords **- *attack tree, exploitation, graphql, injection, time***