Portfolio Reflection

In the final discussion board of this course, the concept of ‘zero trust’ is analyzed from the perspective of a software developer. The first question that is posed is “How does this concept change the way you think about security to incorporate zero trust?”. My answer was simple, this concept only reinforces what I believed to be good quality standard practice for making production code. That is, to always treat every interaction with the system, rather it be from users, databases, or third-party APIs, with the same level of hostility. Specifically, to innately deny access to the program until the entity has been verified, authorized, and authenticated.

The next question pertained to user interactions, namely how they will be impacted once a ‘zero trust’ design is implemented. From the surface, the user will no more aware of the validation process than what is conveyed to them by front-end development. Only if they are not admitted access will they become aware of this protocol. The goal is to have these sorts of checks happen seamlessly based on class privilege within the system. Only allowing users to accomplish and interact with things deemed necessary by the developer. It should never be assumed that the user is only there to use the program as intended.

The final question asks how I, as a developer, intend to persuade other developers that oppose ‘zero trust’ to adopt this concept. A major difficulty that arises when implementing this concept is the complexity of integrating these secure protocols with every interaction that happens internally and externally of the system. My argument is that the time required to make the system secure is far less than that of having to fight exploit ‘fires’ if an attacker manages to find one. With that, the risk of tarnishing the name of the company and the development team is on the line if the team decides to take shortcuts when developing, possibly leading to the leaking of users’/company data.