Darius Quick

2/26/25

8-2 Journal: Portfolio Reflection

Defense in Depth is a strategy/method that is used to make sure that networks are safe by securing important data and information cautiously through systems with multiple layers. If one defense layer were to fail, there would be another layer of defense to handle an attack to the network. Using this method with intended overabundance creates a strong and secure framework that deals with many different methods of network assault.

In this evolving technological world that we live in, network assault is a constant threat from many different types of hackers. Because of this, it is very important to the best type of security put in place to make sure that your frameworks are protected, and your establishment is not put at risk. Unfortunately, there is no easy way to make sure that your framework is protected from every type of network assault. This is even more of a reason why strong secure code is important for networks.

Whereas good protection can take a lot of time, effort, and money the tradeoff is that if you take as many precautions as you can initially, you will have less problems moving forward than if you do not. If you’re network is protecting important information or data that is worth money or effort to assault your system, then it is better to take every precaution to protect your system. And depending on what information and methods you are protecting the best methods of protections and secure code can be chosen. Sometimes the protections are not worth the investment if you don’t have valuable information to protect.

When considering how defense in depth will be utilized it is important to consider many different factors. Those factors include Industry-specific regulations that must be followed, data sensitivity, the architecture of a network, user access management, physical security considerations, application security testing, etc..

The specific technology used, the organization's size and complexity, the sensitivity of data handled, industry regulations, threat landscape analysis, user access controls based on roles, physical security measures, and the level of monitoring and incident response capabilities; essentially, the specific layers of defense chosen and how they are implemented will vary depending on the unique needs of each organization or environment.

Defense in Depth builds the time and intricacy that is necessary to think twice about a network and get rid of any weak links. This is why a DiD approach is regularly taken by software developing companies when designing security for programs.

Don’t leave security to the end means that the security of the program being created should be considered at every sprint, not just once the project has reached completion. This is very important because problems are thought about and prevented as they arise rather than handling every single security flaw once the coding is done, with the possibility of having to go back and correct code.

Some of the steps you can take to prevent threats are applying defense in depth methodologies such as using multilayered security in the code like 2 step verification, user authentication, using security keys, and having try-catch-finally phrases to prevent exceptions in your code. One way I plan to ensure that security is addressed intrinsically and not left an issue until it’s discovered would be to not only check the errors by running and debugging the code in the IDE environment, but to also use Cppcheck to see if there are any other errors or underlying issues with the code on an incremental basis.

The reasons why hackers attack your network can be for three different reasons lawful, monetary, and reputational. Hacker attacks can cause companies to lose money, lose corporate data, monetary data, disturb exchanges, and loss of business agreements. When a company goes through this, they will have to pay handsomely to have the security issue fixed.

Sometimes hackers attack your network to harm your business and the trust that your business clients or associates have for you. This is an attack on your reputation. In any client relationship trust is necessary. Therefore, a hacker attack can cause a corporation loss of clients, deals, and benefits. A network attack can also influence your business associates. You may also accrue fines and administrative assents.

If I were to apply the concept of considering the motive behind a hacker’s attack to my own practice it would be useful in being able to predict how and what the hackers would be aiming to attack within the system. For example, if I were working for a banking company there could be many motives the hacker would have for attacking that system including theft of finances, identity, or to gain kudos or recognition for being able to hack the system. Since that is known I would then seek ways to protect the system from the hacker by protecting that targeted information with multiple layers of defense.

If I were to explain this method to a new developer on my team it would be best to have him view videos of how and why hackers attack the many diverse different systems available so that when he creates code, he also thinks about how the code could be vulnerable and prepare accordingly.

One example of how I could put myself in the shoes of the hacker and understand his motives to prevent an attack on the system would be if they were trying to gain access to a system with user authentication and authorization they would find a way to use SQL injection in order to find a crack in the security system, thereby allowing them to input their own code and gain access to or copy data from the infrastructure. In order to prevent that all the code of the system would have to be secured by making sure that it is protected from injection attacks that would seek to change the initial purpose that the code was meant for.

When it comes to incorporating it to security, the concept of zero trust makes me realize how important security is as a whole to any project.  If zero trust is to be applied, then that means security should be considered just important as the code that is implemented with each sprint (in agile).  Which multiple layers of security will be implemented should be discussed before the code has started to be created in the planning stage of the Software Development Lifecycle.  As a user this policy will make me feel more comfortable working with a company on my project because it will be more secure than if the policy were not in place.

Though my personal belief is that the zero trust policy should be applied as a default to provide the best protection to any program that is created, I cannot force that belief upon anyone.  In order to get developers who oppose zero trust to get on board I would first do research into projects that have incorporated the policy to get positive and negative sides of using it.  I would then bring those results to the developers who oppose it and give them a firm understanding of why it is worth giving it a try.  Whether it continues to be a policy that they will use will depend on the results we get from applying it.

I have a couple of Implementations and recommendations of security policies that should be put into place. For policy development we should clearly define security objectives, scope, and responsibilities within the organization. When we begin the implementation phase of the software development lifecycle, putting policies into practice through training, technology deployment, and ongoing monitoring should be applied.

My recommendations are that there should be regular security assessments that are conducted periodically to identify vulnerabilities and ensure compliance with policies, there should be an incidence response plan in place to address security incidents effectively, and employees should go through awareness training where they are educated about cybersecurity best practices and reporting suspicious activity.