Dylan Coulter February 27, 2025

ID#1896062

C405

**CS405 Module 8 Journal: Portfolio Reflection**

Reflecting on the past two months and how this course has change how I view secure coding has transformed since week one of the course. There have been many large data breaches over the past decade which brings the issue to the forefront for many of us, but this class has expanded on the why and how to protect against attackers and to limit vulnerabilities. It all starts with the adoption of a secure coding standard. Security should always be a top priority during development. Lack in security can have major consequences if something were to happen such as a data breach. When there is a data breach it does not only have an impact on customer or client’s data that was stolen, but also the organization itself. It takes organizations years to gain back trust after a breach happens if they are lucky. Most organizations end up incurring large fines, lawsuits, and negative impacts on their stock. This is why a secure coding standard is so important because with the implementation of a secure coding standard there will be a process to be followed. Code must follow the standard and be compliant and if it does not most likely there will be vulnerabilities that present a risk. These standards would help to mitigate this. This also connects with why it important to not leave security to the end. It is important to always have security in mind and if you are leaving it to the end, you may overlook possible vulnerabilities that most likely will be costly to fix in the future or worse allow for a breach to happen. Attackers are continuously evolving which means what you may have thought was secure yesterday may not be today and that is why it is important to be proactive all the time in identifying any possible risks.

The mitigation of all vulnerabilities that are found will end up saving not just money but also time. When these vulnerabilities are not quickly identified and fixed the is an opportunity that they will be exploited. When we hear of a large data breach this is what usually happens. A vulnerability is found and exploited and after all the damage has been done and the organizing research the incident, they usually find an overlook vulnerability that was not identified and corrected leading to the breach. This is why mitigation is important and usually ends up saving money the long run although most companies may not know the money they are saving until a breach brings to light the cost, legal issues, and loss of clients. To assist with avoiding these risks it is important to run unit test, use static analyzers such as Cppcheck, and fix compiler warnings during the development process. The use of these tools helps to identify vulnerabilities quickly and accurately so they can be address right away. If issues are found and mitigated the final stages such as system testing and deployment will be a lot smoother.

Technology is moving so fast and after Covid we now see workers having the ability to work remote. It is nice to see that remote work can help to save organizations money due saving on supplies and office space, but it poses many cyber security risks. With this a wider network range of data transfers is present and with this it makes it easier for attackers to gain access. With these networks growing so large it because hard to quickly identify who the attacker is and where they are coming from. As seen with many large breaches over the years the long an attacker goes without being noticed or identified the more damage they will cause. This is why zero trust must be implemented. There are many avenues that can be taken with zero trust, and some may be annoying to the user, but if it is important data that is being protected it is necessary. There are a lot of things that can be done on the back end that will be hidden from the user and they would never know such as principle of least privilege. Why give user more access than they need this is just asking for trouble. How would users know they are missing out on something if they never were allowed to use it in the first place. Multifactor authentication is effective and should be use, the only downside is users go through multiple steps the verify they are who they say they are. When you add this with connection timeouts force the user to resign in after a certain period it may become annoying. This is why it is always important to use the right level of security as too much may have a negative impact such as maintenance cost, user frustration, and an overly complex system.

When implementing security policies, it is important to start as soon as possible as the longer you wait the more could go wrong. There are not many benefits in allowing something to be developed and overlooking vulnerabilities or waiting to fix them at a later time. This is why it is important to work with the DevSecOps process. Within this it is important to train software developers on how to keep systems safe especially during a breach. Training developers on the new security policy is important and they should also be taught the common types of attacks that they should be looking out for and how to handle them. Using this will allow for the team to move forward developing more secure code. A focus on Defense-in-Depth is a good way to keep both internal and external systems safe and secure. Defense-in-Depth focuses on the use of multiple layers of security which in turn can help to mitigate potential attackers by making it harder to gain access. The incorporation on encryption and triple-A policies helps to protect data and prevent attackers from accessing it.

.