8-2 Journal: Portfolio Reflection

Prior to taking this course, my concept of programming was linear. I would aim to get from point A to point B as quickly as possible. Security was always someone else’s job. Upon reaching the end of this course I have realized how much I was missing out on. You can not have a multi factored security system in place when the code you use is open to exploit. The best practice is to have security in mind from the beginning and every component added should follow the guidelines set forth to ensure secure coding practices are followed. Imagine writing a program with thousands of lines of code only to have to go back in and redo all your work. Methods would have to be re-written; text input would need to be sanitized, and possibly even having to find and change variables like floats into doubles.

Assessing risks is an important step. You need to be aware of potential pitfalls so that they can be identified, registered, and (hopefully) mitigated. There is an old saying, “an ounce of prevention is worth a pound of cure”, and mitigating risks is just that. Imagine working on a project for a year, only to have someone hack in and delete your entire database of code and data. This would be horrible for you and your company, and it could potentially harm your clients. Raising the type checking level of your compiler will help mitigate some risk before the code is even tested. Include as many flags as you can to check for as many issues as possible. There is a saying from my time in the Army: “The more you sweat in peacetime, the less you bleed in combat.” The saying refers to all the intense training, and how it may be inconvenient, or tiresome, it will help keep you alive when things get real. The same principal applies here; the more you prep your defenses, the less vulnerable you are.

The best policy is a Zero Trust policy. People can be either malicious, careless, or simply do not understand what they are doing. We can not rely on the end user to always enter the proper data into the proper area of the form. Sometimes web portals are visited by malicious actors. Look at what happened with Ashley Madison. This website was hacked and not only were the affairs exposed, but credit card info was exposed as well. Put yourself in the shoes of the offended spouse; not only was your partner unfaithful, but your credit is ruined as well. Equifax was hacked as well because they did not feel as if securing their system was worth their time. I would venture a guess that they never thought they would be hacked. Auto makers will often try to decide if a recall or lawsuit would be cheaper. Ford Motor Company did this with their Escort model. There was a defect in the late 1980’s model where getting hit in the rear could cause an explosion. Ford thought that a lawsuit, or rather settlement, would be cheaper than a recall to fix the problem. Software companies do the same thing. I worked as a tester for a company that knew about a SQL injection vulnerability but ignored it. This company made software that deals with medical records. All these companies trusted that nothing would happen. If there was a Zero Trust policy in place, many of the security issues could have been prevented.

With a well thought out security policy, you have the framework to start adding security features to add to the overall safety of your network. Multiple layers of security such as multi-factor authentication to ensure you are who you say you are, default deny ensuring that someone cannot run wild through the system wreaking havoc. Having an outside security firm analyze your system might be expensive for a small startup, but larger companies should do this to ensure an unbiased assessment along with professional recommendations for improvement. Security only seems expensive prior to a breach. This is somewhat akin to car insurance, where it only pays off after an accident. Other things to think about are training. Employees should be trained in security measures such as physical security. Do not allow entry to people who do not belong. Do not share passwords or leave them written down for others to find. Many times, in the government sector, people wanting to do harm have found the passwords of supervisors who had high levels of access and then deleted or copied data of a sensitive nature. Granted, your average Joe is not storing nuclear launch codes, but imagine if Amazon Web Services were taken over by a disgruntled employee or a black hat hacker. Amazon could suffer substantial damage.

Citations

Seacord, R. C., & Pethia, R. D. (2015). *Secure coding in C and C++*. Upper Saddle River (New Jersey): Addison-Wesley.