Besides building the system requirements, security should be one of the top priorities during development. Without security, it puts everyone at risk, the users, the client, the application, and the development company. There are steps the company can take to stay internally and externally secure. First, adopting a secure coding standard would aid in keeping the code compliant and safe. If the program is not complaint with the coding standard, it is highly likely that it has vulnerabilities. To do this, during the planning phase, the team should go over what programming languages will be used and develop the coding standards for them (Seacord, 2013). Any vulnerabilities can become an issue if attackers find them and exploit them. Next, security should become integrated during the design phase of the development lifecycle. If security is designed into the application, it is less likely to become an issue later in the development and testing phases. Furthermore, the architecture and design of the system drastically impacts how secure the final product is (Seacord, 2013). So, it is not wise to leave security to the end of production. Otherwise, vulnerabilities can get missed during deployment.

Mitigating any vulnerabilities found saves time and money. If vulnerabilities are not fixed when they are identified, they are at risk of being exploited. Mitigation saves money for the company because a system breach can become fatal. Breaches can cause a company to lose clients, have legal issues, and lose millions of dollars. Also, patching and fixing a system after a breach can take a lot of time. So, to avoid these risks, it is important to run unit tests, fix compiler warnings, and use static analyzers during development. These tools help find vulnerabilities quickly so that they can get fixed right away. Furthermore, if issues are found and mitigated, system testing and deployment becomes a lot more efficient.

Today, security is especially important due to the high demand for remote work. Having employees work remotely is great for saving money on office space and supplies, but it comes with a lot of cyber security risks. Having a wider network range for data transfers makes it easier for criminals to gain access to it. Also, since the networks are vaster, it can take a long time to locate where they are. If the attackers don’t get located right away, they can cause a lot more damage. For these reasons, implementing Zero Trust security is very beneficial and essential. Zero Trust is a security model were no user, device, workload, or system should be trusted, no matter where they are located. This model verifies, barricades, and isolates everything in the system. Thus, if an attacker gets into a system, the breach will be more isolated and valuable assets won’t get taken (Check Point Software Technologies, Ltd, 2019).

When it comes to implementing security policies, I would recommend starting as soon as possible. There isn’t really a benefit in letting systems continue to get built with vulnerabilities. When moving towards a DevSecOps process, it is beneficial to train employees on how to keep systems safe during an attack. Also, they should know common types of attacks and how to prevent them, as well as the new security policies. Training will allow them to gain knowledge on what to look for and how to develop secure code. Then, implementing Defense-in-Depth is another great way in keeping systems internally and externally secure. A system having an appropriate number of layers of defense aids in slowing down and stopping breaches. Finally, developing and using Triple-A and Encryption policies can prevent attackers from getting access to sensitive data.

References:

Check Point Software Technologies, Ltd. (2019). *What is Zero Trust Security?* [Video]. YouTube. Retrieved from [What is Zero Trust Security? - YouTube](https://www.youtube.com/watch?v=1D5mg9an19o)

Seacord, R. “Chapter 9. Recommended Practices.” Secure Coding in C and C++, 2nd ed., Software Engineering Institute of Carnegie Mellon, 2013.