Adoption and Assessment of Security Standards

Researching existing security policies, as well as working to develop our own, has helped me to understand the benefits of a robust system of defense using practices, such as defense in depth or adopting secure coding standards. In fact, one of the most straightforward ways to begin coding securely is by adopting a secure coding standard. These are standards developed by experts that, when followed, help developers to produce more secure code. Typically, coding standards define how compliant code should look and provide an explanation for what vulnerabilities can be caused. Not all coding standards will work for your particular work flow. In this case, you can modify and/or create your own coding standards to better fit your individual situation. Unfortunately, security is never fully comprehensive so these are merely attempts at covering as many potential points of attack as possible. When a potential vulnerability is found, it should be assessed for how likely it is to cause further issues or even a security breach; Based on this info, a decision can be made about the benefits and costs of remedying a potential vulnerability. Incorporating coding standards and properly evaluating vulnerabilities when they arise leads to a more secure codebase.

Many standards and rules are based on the principle of zero trust. This policy may seem strict and while it is important to have a strict implementation of your zero trust policy, it’s actually a very logical policy. Essentially, the principle of zero trust means that you should always verify and authenticate data before it’s used. This is accomplished in a variety of ways; Authentication should be performed prior to allowing any user actions and for encrypted data should happen at both ends of transmission. This will require active development of security alongside core features. It’s never too late to start implementing security policies but it’s also never too early. There are risks of more security vulnerabilities if security is left as an afterthought in a project.