Tyler Glover

Alan Spencer

CS-405 Secure Coding

August 24, 2025

Module 8 Journal

As more business and personal activities are moved into technological systems, often connected to the internet, it is paramount that secure coding practices and system architectures are considered at every stage of the software development process. In other words, companies need to transfer from a DevOps development style to a DevSecOps one. Security should not be a consideration only at the end of the development process. If security vulnerabilities are discovered here, it will be expensive, and time consuming to fix them and may require rewriting large parts of the source code. In fact, considering security early can help to make the whole process easier. A test-driven development style that includes security testing can be a valuable organization tool and simplify the coding process by ensuring that code is just good enough to pass the tests. Additionally, a well written secure coding standard can help developers avoid common security pitfalls and create a standardized solution to several coding problems which may decrease development time. Secure coding standards should include principles that guide the policy and individual standards addressing common security issues such as SQL injection or integer overflow.

When developing such a policy, it is important to understand the risks associated with your system as well as the cost versus benefit of mitigation. The first step of understanding risk is understanding why malicious actors may want to target your system. For example, does your system store sensitive user data that could be sold to third parties? If so, your risk assessment needs to include the possibility of a data breach, and modern encryption and hashing techniques need to be considered among the remedies to vulnerabilities. Risks considered should also account for scalability. For example, a new system may be developed for a small business who does not currently do business online; however, it is possible for that business to expand into the digital arena. This may mean that the system under development will be exposed to external threats. When considering security risks, the scalability of a business needs to be considered. When a data breach occurs, the financial and reputation costs are likely to be high; therefore, it is better to give preference for the more secure option considered.

For systems that deal with sensitive data, the Zero Trust security policy should be considered. Many system environments are careful with the say they handle data and system interactions that come from outside the organization; however, as the more rigorous Zero Trust security methodology extends this distrust to interactions that come from within the organization. For example, when accessing system resources, users should have to authenticate their identity. If a password method of authentication is used, an additional authentication method should be added such as an authenticator application. Organizations should also only allow connections from registered devices which are listed explicitly in organization records. These devices should be kept up to date with anti-malware software. This can help keep the system secure if users need to access the system from outside the organization.

When developing new systems, it is recommended that a DevSecOps methodology is used which considers system risks early and often. If the system involves sensitive data, encryption methods are necessary. Zero Trust methodologies should be used, but proper encryption methods will help to prevent negative outcomes in the event a data breach does occur. A system should never be deemed secure. Developers should consider security advancements when the software is in operation, not only during development. If security vulnerabilities are identified during operation, a timely patch must be issued to deal with the flaw.