Courtney Warner  
CS 405   
Journal Module 7  
  
In cybersecurity, understanding the motive behind an attack is just as important as identifying the method. The idea that attackers are not acting randomly but instead have specific goals—whether financial, political, or individual helps us shape stronger defenses and anticipate future threats. As someone entering the field of secure software development, I recognize that staying aware of potential motives behind attacks will help me write more defensive and resilient code.  
I will apply this concept to my practice by always thinking one step ahead when developing or reviewing code. Instead of simply asking, “Can this fail?” I’ll also ask, “Why might someone want this to fail?” or “What could an attacker gain by exploiting this?” This shift in thinking helps me identify vulnerabilities that might otherwise be overlooked. Whether I’m validating inputs, configuring authentication, or managing session data, considering the possible motivations for attacking that specific functionality helps ensure nothing is left to chance. This practice ties directly into threat modeling, which is something I plan to incorporate regularly in my development process.  
To explain this concept to a new developer on my team, I would say: “Imagine your code is a bank vault. You can’t just focus on making sure the door locks; you must think like a thief. Ask yourself, “What’s inside that someone might want? How could they get in? What’s worth targeting?” Framing the idea this way helps shift the mindset from purely technical to strategic. It’s not enough to prevent errors; we must understand why someone would want to break in, and that helps determine where to place our strongest protections.  
One example I will use in my final reflection for Module Eight is the concept of access control and why attackers may try to bypass it. If an attacker’s motive is to steal data or escalate privileges, then even minor access control flaws become high-value targets. In our earlier work on buffer overflow or exception handling, the motive might have been to crash the system or run malicious code. But when we factor in motive, it becomes clearer which areas need stronger attention and monitoring. This awareness strengthens both individual security measures and the overall application design.  
In conclusion, considering the motive behind an attack enhances my ability to develop secure code. It transforms security from a checklist into a mindset. By thinking like an attacker, I can better defend the systems I help build.