Nathan Quirk  
Article Review 4  
“Bridging the Gap between Software Development and Information Security”

In “Bridging the Gap between Software Development and Information Security,” Kenneth R. Van Wyk and Gary Mcgraw suggest ways for firms to incorporate a security mindset into their development practices. The authors feel that these concerns are not taken seriously enough, and that the division between the security and software development worlds is an unhealthy one.

Right off the bat, the authors address the lack of hardened security experts within the development domain. These experts have had years of intrusion prevention and mitigation experience, which cannot be replicated. Having them work hand-in-hand with developers is the only way to infuse their expertise into a project.

In addition to employing security professionals, the authors focus on best practices or “touchpoints” that make software more resistant to attack. The first of these is *abuse cases*. Here the idea mirrors use cases, or user stories in agile-ese. In them, developers consider how some aspect of their software might be used in a manner not consistent with the designer’s intentions, and what impact that might have on system security. In this way, developers can attempt to foresee potential disasters before they happen. Once an abuse case is identified, it proceeds much the same way as a use case: the issue generates a new requirement, and fix is developed and then tested.

Wyk and Mcgraw then advocate for accuratebusiness risk analyses to determine what impact a security breach could have on the firm. This helps to garner support for the necessary expenditures of time and money needed to properly secure software. Without this step, efforts to bolster project defense are likely to be carried out half-heartedly.

Projects also need to be evaluated from a wider perspective. Assessing the risk potential from an architectural standpoint is also important. Developers need to consider how the entire application’s design aligns with security concerns. As the article puts it: “at least 50 percent of all security defects are architectural in nature.”

Security testing should mirror functionality testing, much like use and abuse cases. Here tests are run to determine whether safeguards function as they are supposed to. Testing can be “positive” or “adversarial.” In other words, you can think like a good guy, and test the defenses or attempt to break the system by trying to undermine the architectural constraints in place.

While it is useful to identify these practices, implementing them is another story. Here Wyk and Mcgraw suggest some ways that software developers and security professionals can incorporate these ideas by working together.

Abuse cases – collaboration between security professionals and developers is critical here. Developers don’t know what they don’t know – security people have seen intrusions first hand and can bring those experiences to bear when thinking of attack vectors. Developers may balk at first and some convincing might be required. However, the authors warn against overstating the severity of risks, instead, let the facts do the talking.

Business risk analysis - bringing the business stakeholders here is key to getting everyone on the same page. Only they can answer questions that might illuminate the cost of a security breach or the resulting downtime. Seasoned security professionals also contribute by relating their past experiences with intrusions and how they were handled. They can help to flesh out precisely what may be at stake for the developers should they fail to consider vulnerabilities.

Test Planning – although this is usually done by an in-house quality assurance department, security professionals also have an opportunity to shine here. Instead of creating unrealistic and flimsy tests, they can get into the mind of an attacker and effectively challenge security protocols, thus hardening the final product.

Penetration Testing - here the security professional’s role is obvious. Being on the front lines of attack means that these people have the best chance to seek out and illuminate any lurking holes in the fortifications.

Finally, Deployment and Operations - here the security team can be useful in setting up and deploying the application so that all the careful attention isn’t wasted on a faulty rollout.