### 

### Southern New Hampshire University

3-2 Journal: Reflection

David Vega

### Dr. Vivian Lyon

### CS-305 – Software Security

01/23/2023

# 3-2 Journal: Reflection

My role as a developer in solving security concerns would be to work with IT security to identify common vulnerabilities. It is often called the “blame game” because in one hand IT security claims they can’t trust developers to write secure code while on the other hand, developers say they aren’t given proper guidance (Security Intelligence).

It is the developers’ job to get the application out quickly. Vikram Kunchala a U.S. lead for Deloitte’s cyber cloud practice says in an article written by Mark Stone a Hubspot-certifed content marketing writer and called “Who’s Responsible for Application Security?” that,

*“If I’m a developer, my charter is to get functional and good quality code out, because there is speed time to market…so security is probably not on top of my mind.”*

I would have to agree with this statement to some extent. Since I’ve been learning about software development (almost 10 years now), I’ve rarely thought of security in my applications…well, other than authentication and input validation when needed. However, much like technology is constantly evolving, I suppose it’s only fitting if developers evolve too, to expand their skill sets to think of security much like they think of “functional and good quality code.” So, does the original statement still stand? Has a developer’s charter expanded to include security in determining what constitutes “functional and good quality code?” I think it would involve having conversations with appropriate groups to come up with a cohesive development plan that includes security at most if not all stages of SDLC. This would include the tools and technology stacks that we utilize in projects. If open-source technologies are used, the conversation should be around who supports it because if there is no support for maintaining it, then it might not be worth including it in the tech stack (Jeganathan, 2019). The responsibility to maintain would fall on the organization using it and that could put a strain on budgetary resources.

Security should be at the forefront of software development and not an afterthought. In the same way developers think about functionality, security should also run parallel. Security in the SDLC can even start as early as identifying the tools being used. Things like securing source control, setting up environments, ensuring proper access controls, etc.

The article suggests that following the below plan to secure the entire DevOps life cycle is not a one size fits all solution but that it could provide a baseline to work with when implementing in your own groups (Jeganathan, 2019):

* High-level rapid risk assessment and evaluating threat models.
* Plan and secure DevOps lifecycle tools (i.e. – Gitlab, Azure DevOps, etc.)
* Secure keys and privileged service accounts, API keys, etc.
* Define access controls for all groups

I feel this plan is a good baseline. It covers a lot of the local security items upfront that we as a group can control. There should be no question as to why we can’t secure our own environments by putting some protections in place. Not all vulnerabilities come from outside.

# References

Jeganathan, S. (2019). DevSecOps A Systemic Approach for Secure Software Development. *ISSA Journal*, 20-27.

Stone, M. (2023, January 23). *Application Security Developers Who is Responsible?* Retrieved from SecurityIntelligence: https://securityintelligence.com/articles/application-security-developers-who-is-responsible/