Victor Feight

10/19/2021

Adoption of a secure coding standard, and not leaving security to the end

It is important to adopt a secure coding standard early on (requirements stage) of the SDLC to ensure that secure mechanisms are in place from the start. As explained in my presentation, we can do this keeping servers under lock and key on premises with only priveleged users able to access them. In the Supplier cloud, ensure privacy level agreements and service level agreements are in place. Without our own private cloud, establish a baseline of deny-by-default configurations and defense in depth through external pentests and tried and true encryption mechanisms, as well as enabling Firewall and priveleged account management (PAM), and Monitoring for unauthorized changes and access through logging. Vulnerability scans, patch management techniques, and updates of vulnerable apis can be integrated into the CI/CD pipeline and automated. By protecting our host security, we protect the critical assets of the system and data that attackers seek. Lastly, training of the team on the layered defense compliance protocols is critical.

Evaluation and assessment of risk and cost benefit of mitigation

The Systems Sciences Institute at IBM reported that it cost 6x more to fix a bug found during implementation than to fix one identified during design. Furthermore, according to IBM, the cost to fix bugs found during the testing phase could be 15x more than the cost of fixing those found during design (Cost to fix bugs)

Zero trust recognizes that trust needs to move beyond network perimeter, assuming we trust no one and verify everyone to allow correct user resource priveleges. No devices, users, workload, or system trusted by default inside or outside the security perimeter according to zero trust paradigm. Identity should not rely on only username/pass, but access control to each device (every device connected should be treated as a threat vector, according to the video “What is Zero Trust Security?”

Zero trust

Implementation and recommendations of security policies

In Pre-production, One can automate a selection of Unit tests (DESIGN STAGE) as well as static analysis tools (VERIFY AND TEST) in the CI/CD pipeline (BUILD) thereby providing secure-fail/pass criteria on builds.

By documenting possible or known threats (by focusing on data flow) we can address them, rank them, prioritize assessment of mitigation (ASSESS AND PLAN).

In production, things become more expensive to fix. We practice defense in depth with configuration of security settings, monitor logs, respond to and block attacks, and maintain a secure and stable baseline that we can return to if compromised.