Defense in depth (DiD) is explained by our textbook as the practice of managing “risk with multiple defensive strategies so that if one layer of defense turns out to be inadequate, another layer of defense can prevent a security flaw from becoming an exploitable vulnerability” (Seacord, 2013, p. 181). To implement a good DiD strategy one must consider the sensitivity of data being protected, the cost of each layer of defense, the added difficulty in implementation and ongoing management as well as the impact to end users. Data such as banking or payment information and protected health information must be protected well enough to meet or exceed expectations set by the government in the location the company will be working. Sensitive data has stricter requirements than less sensitive data like the high score on a game application and thus warrants the additional cost that added layers may bring. The question of how deep is too deep is as a result heavily dependent on how sensitive the data in question is. The depth needs to be sustainable, effective and meet all applicable requirements.

The cost of each defensive layer includes not only the monetary cost of the service, software, and hardware, but also the cost of the support team that may be needed to manage the layer and ensure stability moving forward. For example, large healthcare corporations frequently have a team dedicated to managing the firewall for the network including any tunnels or connections that may need to bypass said firewall to allow critical functionality. The cost of employing said support team needs to be assessed as part of the cost of maintenance. An additional item to consider is the added time to perform tasks. Additional levels of security can come with a measurable slowdown in performance, if the security is warranted by the sensitivity of the data, this can be an acceptable tradeoff unless it is a system where quick response is vital, like hospital cardiac systems or emergency response systems.

Systems that include less sensitive data may not have as many requirements for the data’s protection but every precaution the company and software can sustainably utilize should be utilized. When systems are infiltrated even if no data is taken and the services are just interrupted users will lose trust in the company. This means safe guarding the system protects the reputation of the company. Furthermore, since most applications have a way of monetizing the application to generate income for the company many frequently store some type of sensitive account data. Loss of trust in users can result in monetary loss for the company as users transition to a perceived safer competitor.

Defense in depth frequently includes things like frequent updates of software to get patches in places as soon as feasible. It also includes firewalls, antivirus software, multi-factor authentication, the application of lowest privilege needed principals, physical security, auto-timeout for inactivity and much more. It also includes using best secure practice in initial development and sanitizing untrusted data before letting it interact with the system. Defense in depth can include input checks, parameterization, security/authorization calls in high risk functionality, security testing. These layers all work together to ensure the data is as safe as we can plausibly make it. It should also include a security team that can detect infiltration and respond to limit damage as quickly as possible. These items all come with cost of time, money, performance, and ongoing support needs but are vital to sustained success in the industry. A serious discussion on the sensitivity of the data stored, a budget for security and the needs of the users/system are required to determine how deep is deep enough, and how deep is too deep when creating the layers of defense for a system or solution.

References

Seacord, R. C. (2013). *Secure Coding in C and C++* (2nd ed.). Pearson Technology Group. <https://mbsdirect.vitalsource.com/books/9780132981972>