**Week4 Reflection on Cybersecurity**

Hang Yang

IS 5403: Cybersecurity

Prof. Jaime Martinez

April 06, 2024

**Week3 Reflection on Cybersecurity**

This week we learnt database security and vulnerability management from Dr. Mclver’s video. We have noticed that the security concerns around data storage arises for quite a long time. In the database security side, Prof. Samuel suggested the first thing we think about is the possibility of segregation of your database server from other components in the network under budget coverage. The second thing we should focus on is monitoring and educating user behaviors, particularly those of admin users, to prevent granting excessive permissions and to mitigate the risk of SQL injection through input sanitization techniques. Another effective approach to safeguard the data storage is to add system auditing logs around operations against the data storage application. This could provide insights on how to identify root causes and provide potential recovery solutions. Of course, the mentioning of encryption of the confidential or critical data is also helpful. Among all of these safety measures, Prof. Samuel highlighted the importance of segregation and disagreed to put all the critical components into DMZ. Because this could inadvertently attract hackers and increase the risk of unauthorized access due to potential human errors or unforeseen incidents. Instead, he advocated for physical segregation as a proactive measure to minimize such risks.

The second video introduced vulnerability management. Vulnerability management is a critical aspect of cybersecurity, as underscored by recent incidents such as the 2021 Chrome Zero-day vulnerability and the 2020 Zoom and Apple iOS vulnerabilities. These events serve as stark reminders of the real-world consequences of neglecting vulnerability management practices. While the ideal scenario would involve writing flawless, bug-free code, the reality is that software development inevitably introduces the presence of bugs. These bugs may originate from various factors, including limited knowledge, inherent flaws in programming languages or libraries, or unforeseen circumstances. To safeguard against vulnerability attacks, several proactive measures can be taken. Firstly, we could maintain an inventory of all software, operating systems, applications, and hardware components, ensuring they are regularly updated. Secondly, we should provide training and education to internal users on security best practices and potential vulnerabilities. Thirdly, we could establish an incident dashboard and response team capable of promptly addressing and mitigating any vulnerabilities or security breaches as they arise.

This week's Coursera course delved into fundamental concepts in networking, covering topics such as IP addresses, domain name resolution, and the OSI model's seven layers. Additionally, it provided insights into data structures and storage systems, including SQL databases, NoSQL databases, data warehousing, and various database roles. One of the most valuable takeaways for me was learning how to prevent injection attacks, such as command injection and SQL injection, from a developer's perspective. The course offered valuable suggestions and principles, such as utilizing built-in functions whenever possible, sanitizing user inputs, employing mapping tables, and using prepared statements and parameterized queries. These practices are widely recognized as best practices in the industry and are crucial for enhancing application security. Additionally, the final project introduced the use of tools like Snyk, which can help identify, prioritize, and address security vulnerabilities seamlessly integrated with many source code management tools.

**References**

Kaspersky(n.d.). What is a Zero-day Attack? - Definition and Explanation. Kaspersky. Retrieved from

https://usa.kaspersky.com/resource-center/definitions/zero-day-exploit