Book Questions
Chapters 14
Devayan Mandal

14.1 Explain how the complementary strategies of resistance, recognition, recovery, and reinstatement may be used to provide system resilience.

Ans: The complementary strategies of resistance, recognition, recovery and reinstatement maintain system resilience in the following ways:

a. **Resistance**: This factor plays an important role in system resilience because it reduces the frequency of malfunction of the system. Theoretically, the system should be able to provide complete resistance to all forms of malfunction (for example malfunctioning due to cyber-attacks). But practically, complete resistance against attacks may not be viable and hence the highest resistance put forth by a system is what resilience engineers strive for.

b. **Recognition**: This factor is usually seen first while designing a resilient system because the system should be able to recognize threats or attacks, which might lead to system malfunction in a timely and appropriate manner. State-of-the-art information and process security systems and 24/7 surveillance of external attacks ensures that a threat can be recognized before it penetrates the system.

c. **Recovery**: This factor addresses the phase wherein the system recovers its core processes after system malfunction due to an external attack. The recovery phase ensures that the processes critical to the system are given primary restoration importance.

d. **Reinstatement**: This factor addresses the phase after the system has reached stability post-recovery. It is ensured that all system processes are running fully as expected and that no effects of the threat further persist.

14.5 Suggest three defensive layers that might be included in an information system to protect data items from changes made by someone who is not authorized to make these changes.

Ans: Protection of data items from changes made during any type of unauthorized access is critical. Defensive layers may need to be implemented in addition to the primary login process. One layer of defense might be the use of checksums. If there is unauthorized modification in a component of system data, there will be a change in the checksum related to the modified data component too. Another layer of defense would be automated checks for malware on system startup. An effective but expensive layer of

defense would be incorporating additional authentication information via finger print or eye scans with the existing login process. This would ensure a superior level or protection.

For all of the layers of defense, it is crucial to maintain backup copies of information and data (preferably stored locally) so that the system components can be restored if there happens to be unauthorized data modification.

14.7 Suggest how the approach to resilience engineering that I proposed in Figure 14.9 could be used in conjunction with an agile development process for the software in the system. What problems might arise in using agile development for systems where resilience is important?

Ans: The agile software development model can be used to develop a system resilient to external attacks. As we observe in Fig. 14.9 (Page 412) the model of resilience engineering comprises of several process layers. An agile development process can be used to implement and test a layer of protection for each iterative development in the system's software. For example, every time a component of the system's software is updated or a new software component is purchased, a set of iterative resilience engineering tests may be carried out to ensure that the existing layers of protection are still satisfactory.

One of the main purposes of resilience engineering is to protect critical system components from exposure to external threats. For this case, the agile development model faces a drawback because it's design procedures do not primarily focus on identifying and acquiring the critical system components first. The agile process leaves room for improvement in each iterative step and so, allocation of critical system components may be finalized only after design completion of the software system.