ITIS6200 EXERCISE: 04-Q3&Q4

Amaan syed (asyed15@uncc.edu)

Question 3:

A computer system provides protection using the Bell-LaPadula policy. How would a virus spread if.

(a) the virus were place on the system at system low (the compartment dominated by all other compartments)? Justify your answer.

Computer viruses propagate through write operations. According to the Bell-LaPadula model, a process can write to any item within the category (compartment) that governs the category (compartment) in which the process resides. Each category (compartment) controls the system at its lowest security level, allowing a system-low process to write to any resource within any category (compartment). In other words, a computer virus could spread even when assigned a low system privilege.

(b) the virus were place on the system at system high (the compartment that dominates all other compartments)? Justify your answer.

Within the Bell-LaPadula Model, a process is permitted to write to any object within a category (compartment) governed by the category (compartment) containing the process itself. Since each compartment across the system is under the control of the system-high category (compartment), a computer virus is restricted from spreading to other compartments. However, it could still propagate among objects within the same classification at the same level (system high in this case).

Question 4:

Classify the following vulnerabilities using the RISOS model. Assume that the classification is for the implementation level. Justify your answer.

(a) The presence of the “wiz” command in the sendmail program (see section 20.2.8).

Sendmail programs using the 'wiz' command are inadequate for identifying, verifying, or authorizing an attacker. By guessing the permissions of the mail-sending server, an individual could execute commands with the privileges of that user, leading to unreliable user identification.

(b) The failure to handle the IFS shell variable by loadmodule (see section 20.2.8).

Since the source code is available, this concept can be tested by disassembling the load module executable. The approach requires modifying the IFS variable to include '/', clearing PATH and IFS, and renaming the initial small program originally in 'ld'—all without altering the source code. Navigate to 'bin' to initiate the load module, which results in an effective UID of 0 for the process, confirming successful execution. This outcome reveals a fundamental flaw in generalization: environments in which privileged programs run must not rely solely on subprogram trustworthiness due to issues with subprocesses and inherited environment variables. This illustrates a weakness in the parameter validation process.

(c) The failure to select an Administrator password that was difficult to guess (see section 20.2.9).

Within the RISOS framework, the inability to set a hard-to-guess Administrator password is classified as an Elevation of Privilege security risk. This classification arises from the likelihood that an attacker could guess the password, gain elevated access, and exploit system resources without authorization. By selecting a weak or easily guessed password, the administrator inadvertently grants attackers the potential to escalate privileges and execute harmful actions on the system. Thus, an implementation-level Elevation of Privilege vulnerability results from the failure to establish a sufficiently secure Administrator password.