Introduction to Computer Security, exercise 3, April 12-16, 2010

1. Introduce the malware *rootkit*. What malware class does it belong to, or is it its own class? How is it detected and destroyed?

2. Check ten most recently updated vulnerabilities published by CERT (http://www.kb.cert.org/vuls/byupdate?openstart=1count=10). Make a summary and analyse with every vulnerability, if some of the implementation or management rules have been broken.

3. Read Critical Control 3: Secure Configurations for Hardware and Software on Laptops, Workstations, and Servers (http://www.sans.org/critical-security-controls/) from SANS institute's web pages, reference 20 Most Critical Security Controls. Make a summary.

4. One program used setuid-to-root privileges when performing some operations. Someone observed that it could equally well implemented as a server, in which case the program would authenticate the user, connect to the server, send the command and role, and then let the server execute the command.

   a) What are the advantages of using the server approach rather than the single program approach?

   b) If the server responds only to clients on the local machine, using interprocess communication mechanisms on the local system, which approach would you use? Why?

   c) If the server were listening for commands from the network, would that change your answer to part b). Why or why not?

   d) If the client sent the password to the server, and the server authenticated, would your answers to any of the three previous parts change? Why or why not?

5. The canary for StackGuard simply detects overflow that might change the return address. This exercise asks you to extend the notion of a canary to buffer overflow.

   a) Assume that the canary is placed directly after the array, and that after every array access canary is checked to see if it has changed. Would this detect a buffer overflow? If so, why do you think this is not suitable for use in practice? If not, describe an attack that could change a number beyond the buffer without affecting the canary.

   b) Now suppose that the canary was placed directly after the buffer but -like the canary for StackGuard- was only checked just before a function return. How effective do you think this method would be?