Such utilities are relatively crude; application-level timeouts are preferable to the blunt object approach of operating system-level logoff utilities or generic screen-lock programs. Using application timeouts, a program can periodically branch to a security screen for re-authentication. A security screen can ask for a password or for other authentication information such as questions from a personal profile. Best of all, such application-level functions, being programmed in by the development team that knows how the program will be used or is being used in practice. To identify inactivity, one uses a timed terminal read. A function can monitor the length of time since the last user interaction with the system and set a limit on this inactivity. At the end of the timed read, the program can branch to a special reauthentication screen. Filling in the right answer to a reauthentication question then allows the program to return to the original screen display. Since programmers can configure reauthentication to occur only after a reasonable period of inactivity, most people would not be inconvenienced.

A really smart program would actually measure response time for a particular entry screen for a particular user and would branch to the security screen only if the delay were much longer than usual; e.g., if 99% of all the cases where the John accessed the customer-information screen were completed within 5 minutes, the program would branch to the security screen after 5 minutes of inactivity. In contrast, if Jane took at most 10 minutes to complete 99% of her accesses to the employee-information screen, the program would not demand reauthentication until more than 10 minutes had gone by.

In summary, an ideal timeout facility would be written into application program to provide

- A configurable time-out function with awareness of individual user usage patterns;
- Automatic branching to a security screen for sophisticated reauthentication;
- Integration with a security database, if available;
- Automatic return to the previous (interrupted) state to minimize disruption of work.

* * *

M. E. Kabay, PhD, CISSP is Associate Professor in the Department of Computer Information Systems at Norwich University in Northfield, VT. Mich can be reached by e-mail at <mkabay@compuserve.com>. He invites inquiries about his information security and operations management courses and consulting services. Visit his Web site at <http://www.mekabay.com/index.htm> for papers and course materials on information technology, security and management.

Copyright © 2001 M. E. Kabay. All rights reserved.
Permission is hereby granted to *Network World* to distribute this article at will, to post it without limit on any Web site, and to republish it in any way they see fit.