This is the second in a series of four articles responding to a reader's request for information on security for programmers.

>Do any current computer programming languages give programmers tools with which to implement security best practices in their code?<

All computer languages allow you to write code as well as you can <smile>. I think that strongly-typed languages may offer better constraints on programmers, but the essential issue is that the programmers continue to think about security as they design and implement code. Java does include provisions for limiting access to resources outside the "sandbox" reserved for a process, as described in the books by Felten and McGraw.

>Is there any such thing as security best practices for computer programmers?<

In a sense, though not, as far as I know, in any codified form. There are recommendations on security-related aspects of programming in most general security textbooks; see for example Stallings.

In addition to designing security into a system from the start, I can think of some obvious guidelines that can apply:

* Impose strong identification and authentication for critical and sensitive systems in addition to the I&A available from the operating-system; ideally, use token-based or biometric authentication as part of the initialization phase of your application.

* Document your code thoroughly, including using data dictionaries for full definition of allowable input and output to functions and allowable range and type of values for all variables.

* Use local variables, not global variables, when storing sensitive data that should be used only within a specific routine; i.e., use the architecture of the process stack to limit inadvertent or unauthorized access to data in the stack.

* Re-initialize temporary storage immediately after the last legitimate use for the variable, thus making scavenging harder for malefactors.

* Limit functionality in a specific module to what is required for a specific job; e.g., don't use the same module for supervisory functions and also for routine functions carried out by clerical staff.

* Define views of data in databases that conform to functional requirements and limit access to sensitive data; e.g., the view of data from a medical-records database should exclude patient identifiers when the database is being used for statistical aggregation by a worker in the finance department.
More on this subject in the next column.

Mich Kabay can be reached by e-mail at <mkabay@atomictangerine.com>. He invites inquiries about a wide range of information security courses and INFOSEC consulting services that he and his colleagues at AtomicTangerine would be delighted to deliver to your employees at your site and at your convenience. For Web-based or CD-ROM online training in security from our INFOSEC University project, see <http://infosecu.com>.

For information about AtomicTangerine, visit <http://www.atomictangerine.com>.

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.