I am delighted to report that Microsoft is getting religion. The following new text is heartening grounds for optimism:


According to the biographical notes, “Michael Howard is a founding member of the Secure Windows Initiative team at Microsoft, a team with the goal of convincing designers, developers, and testers that they need to deliver secure systems.” “David LeBlanc, PhD, currently works in Microsoft’s Trustworthy Computing Initiative and has been part of Microsoft’s internal network security group as a tools developer and white-hat hacker. Prior to joining Microsoft, he led the team that produced the Windows NT version of Internet Security Systems’ Internet Scanner.”

The books opens with a good introduction that explains the increasing importance of security (I wish this principle had been accepted earlier at Microsoft) and correctly defines “secure code” as “code that is designed to withstand attack by malicious attackers. Secure code is also robust code.” The book is intended to teach the reader “to design, write and test application code in a secure manner” by being “relentlessly practical.” The book is aimed application designers, programmers, testers and documenters for Windows 32-bit platforms and for the Web.

Part I – Contemporary Security – opens with Chapter 1, “The Need for Secure Systems,” where the authors summarize the consequences of bad security in production programs and suggests practical approaches to convincing your colleagues and bosses that shipping rotten software is not a good idea in the long run. Chapter 2, “Designing Secure Systems, starts by pointing out two widespread errors in program design and coding: (1) simply forgetting about the security of their code (or adding irrelevant security features that don’t mitigate the fundamental flaws of the design); (2) applying security functions to the code after it’s been designed and coded. Their discussion of security principles is encouraging:

* Establish a security process
* Define the product security goals
* Consider security as a product feature
* Learn from mistakes
* Use least privilege
* Use defense in depth
* Assume external systems are insecure
* Plan on failure

* Fail to a secure mode

* Employ secure defaults

* Remember that security features != secure features

* Never depend on security through obscurity

The chapter continues with “Security Design by Threat Modeling” and “Security Techniques.” In addition to reading this chapter, readers will benefit from reading Chapter 3 (Using a “Common Language” for Computer Security Incident Information”) by John D. Howard & Pascal Meunier; and Chapter 5 (Toward a New Framework for Information Security) by Donn B. Parker in the _Computer Security Handbook, 4th Edition_ to expand their horizons in thinking systematically about security threats and vulnerabilities.

For lack of space, I won’t go into further detail about Howard and LeBlanc’s new text. The other section and chapter titles are as follows:

Part II: Secure Coding Techniques
- 3 Public Enemy #1: The Buffer Overrun
- 4 Determining Good Access Control
- 5 Running with Least Privilege
- 6 Cryptographic Foibles
- 7 Storing Secrets
- 8 Canonical Representation Issues

Part III: Network-Based Application Considerations
- 9 Socket Security
- 10 Securing RPC, ActiveX Controls, and DCOM
- 11 Protecting Against Denial of Service Attacks
- 12 Securing Web-Based Services

Part IV: Special Topics
- 13 Writing Secure .NET Code
- 14 Testing Secure Applications
- 15 Secure Software Installation
- 16 General Good Practices

I also very much appreciate the CD packaged with the book; it includes a complete electronic copy of the text – a terrific boon for teaching because of its searchability. I will definitely consider this text for my next university courses on quality assurance and advanced programming. Congratulations to the authors and to Microsoft. Now let’s hope these principles actually get used at Microsoft and elsewhere despite decades of violation of all of these best practices.

* * *

NEW! 18-month online Master of Science in Information Assurance offered by Norwich University; see <http://www.norwich.edu/msia> for full details.

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>; Web site at <http://www.mekabay.com/index.htm>.

Copyright © 2002 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.