THE EXPERT'S VOICE™

Jason Bock,
Pete Stromquist,
Tom Fischer,
and
Nathan Smith

.NET Security

Learn the basics of cryptography and security
as they are implemented in the .NET Framework

Understand the .NET cryptography classes
and XML encryption and signatures

Learn about role access security and secure remoting

eveloper Series

Apress™

.NET Security

JASON BOCK, PETE STROMQUIST,
TOM FISCHER, AND NATHAN SMITH

Apress™

ISBN (pbk): 1-59059-053-8
Printed and bound in the United States of America 12345678910

Technical Reviewers: Brock Allen, Robert Knudson, and Chris Sells
Editorial Directors: Dan Appleman, Peter Blackburn, Gary Cornell, Jason Gilmore, Karen Watterson, John Zukowski
Managing Editor: Grace Wong
Copy Editor: Ami Knox
Compositor: Impressions Book and Journal Services, Inc.
Indexer: Valerie Haynes Perry
Cover Designer: Kurt Krames
Manufacturing Manager: Tom Debolski
Marketing Manager: Stephanie Rodriguez

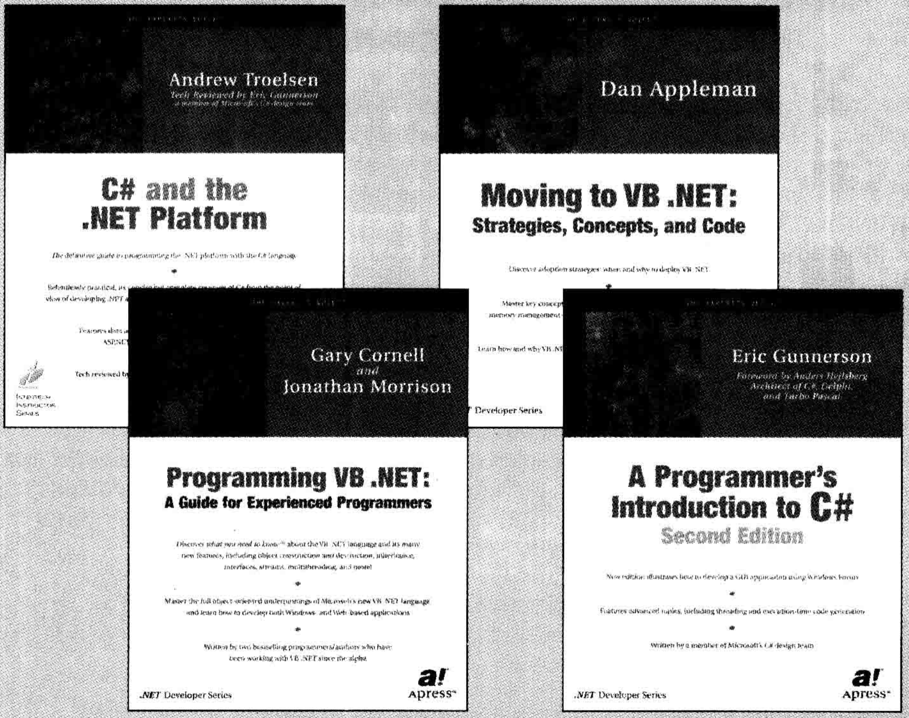
Distributed to the book trade in the United States by Springer-Verlag New York, Inc., 175 Fifth Avenue, New York, NY, 10010
and outside the United States by Springer-Verlag GmbH & Co. KG, Tiergartenstr. 17, 69112 Heidelberg, Germany.

In the United States, phone 1-800-SPRINGER, email orders@springer-ny.com, or visit http://www.springer-ny.com.
Outside the United States, fax +49 6221 345229, email orders@springer.de, or visit http://www.springer.de.

For information on translations, please contact Apress directly at 2560 Ninth Street, Suite 219, Berkeley, CA 94710. Phone: 510-549-5930, Fax: 510-549-5939, Email: info@apress.com, Web site: http://www.apress.com

The information in this book is distributed on an “as is” basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author nor Apress shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

The source code for this book is available to readers at http://www.apress.com in the Downloads section. You will need to answer questions pertaining to this book in order to successfully download the code.



Andrew Troelsen
C# and the .NET Platform
Dan Appleman
Moving to VB .NET:
Strategies, Concepts, and Code
Gary Cornell
and
Jonathan Morrison
Programming VB .NET:
A Guide for Experienced Programmers
.NET Developer Series
a! Apress
Eric Gunnerson
A Programmer's Introduction to C#
Second Edition
.NET Developer Series
a! Apress

a!™ Apress™ *books for professionals by professionals™*

About Apress

Apress, located in Berkeley, CA, is a fast-growing, innovative publishing company devoted to meeting the needs of existing and potential programming professionals. Simply put, the "A" in Apress stands for *"The Author's Press™"* and its books have *"The Expert's Voice™".* Apress' unique approach to publishing grew out of conversations between its founders Gary Cornell and Dan Appleman, authors of numerous best-selling, highly regarded books for programming professionals. In 1998 they set out to create a publishing company that emphasized quality above all else. Gary and Dan's vision has resulted in the publication of over 50 titles by leading software professionals, all of which have *The Expert's Voice™*.

Do You Have What It Takes to Write for Apress?

Apress is rapidly expanding its publishing program. If you can write and refuse to compromise on the quality of your work, if you believe in doing more than rehashing existing documentation, and if you're looking for opportunities and rewards that go far beyond those offered by traditional publishing houses, we want to hear from you!

Consider these innovations that we offer all of our authors:

- **Top royalties with *no* hidden switch statements**
  Authors typically only receive half of their normal royalty rate on foreign sales. In contrast, Apress' royalty rate remains the same for both foreign and domestic sales.

- **A mechanism for authors to obtain equity in Apress**
  Unlike the software industry, where stock options are essential to motivate and retain software professionals, the publishing industry has adhered to an outdated compensation model based on royalties alone. In the spirit of most software companies, Apress reserves a significant portion of its equity for authors.

- **Serious treatment of the technical review process**
  Each Apress book has a technical reviewing team whose remuneration depends in part on the success of the book since they too receive royalties.

Moreover, through a partnership with Springer-Verlag, New York, Inc., one of the world's major publishing houses, Apress has significant venture capital behind it. Thus, we have the resources to produce the highest quality books *and* market them aggressively.

If you fit the model of the Apress author who can write a book that gives the "professional what he or she needs to know™," then please contact one of our Editorial Directors, Gary Cornell (gary_cornell@apress.com), Dan Appleman (dan_appleman@apress.com), Peter Blackburn (peter_blackburn@apress.com), Jason Gilmore (jason_gilmore@apress.com), Karen Watterson (karen_watterson@apress.com), or John Zukowski (john_zukowski@apress.com) for more information.

About the Authors

Jason Bock is an instructor and consultant for Intertech, Inc. He has worked on a number of business applications using a diverse set of tools and technologies such as VB, COM, and Java. He is also the author of *CIL Programming: Under the Hood*^{TM} *of .NET* and *Visual Basic 6 Win32 API Tutorial*, and has written articles and given presentations on technical development issues within VB. He has a bachelor's degree and a master's degree in electrical engineering from Marquette University.

When he's not developing programs, writing books, or giving presentations, Jason enjoys spending time with his wife Liz, golfing, biking, reading, listening to and playing music, and watching *The Simpsons* whenever he can. Visit his Web site at `http://www.jasonbock.net`.

Pete Stromquist is a consultant at Magenic Technologies (one of the nation's premier Microsoft Certified Solution Providers), specializing in Web-enabled application development using Microsoft tools and technologies. Pete has spent the last several years architecting and developing the following types of applications: intranet content management, Web-enabled training and testing software, B2B and B2C e-commerce, and Web-based telemetry and logistics. Pete has complemented his VB skills with several other technologies such as XML, XSL, COM+, IIS, ASP, and of course .NET. Pete also enjoys teaching and giving presentations on .NET technologies. Pete has a mechanical engineering background, receiving his bachelor of science from the University of Minnesota.

Tom Fischer's career spans a broad range of technologies with some of the most prestigious consulting firms in the Twin Cities. His certifications include the Sun Certified Java Programmer (SCJP), Microsoft Certified Solution Developer (MCSD), and Microsoft Certified Database Administrator (MCDBA). And as a Microsoft Certified Teacher (MCT), Tom also helps teach other developers about the latest Microsoft .NET tools and technologies.

Nathan Smith is a consultant with Spherion in Scottsdale, AZ. He holds almost every Microsoft acronym possible (all but MCT) and specializes in the development of and conversion to Web-enabled applications. Prior to the first beta release of C#, he focused primarily on Visual Basic development, which he's been involved with for approximately six years.

About the Technical Reviewers

Chris Sells is an independent consultant, specializing in distributed applications in .NET and COM, as well as an instructor for DevelopMentor. He's written several books, including *ATL Internals*, which is in the process of being updated for ATL7 as you read this. He's also working on *Essential Windows Forms* for Addison-Wesley and *Mastering Visual Studio .NET* for O'Reilly. In his free time, Chris hosts the Web Services DevCon (scheduled for November 2002 this year) and directs the Genghis source-available project. More information about Chris and his various projects is available at `http://www.sellsbrothers.com`.

Brock Allen is a trainer for DevelopMentor as well as an independent consultant specializing in .NET and ASP.NET. He resides in Medfield, MA, with his wife and two dogs. He can be contacted at `ballen@develop.com`.

Robert Knudson has been active in commercial .NET development for a year and a half. This work has included framework development for Microsoft and involvement on a large ASP.NET site. In addition, Bob has served as an editor for various .NET texts. He has been a software development consultant for eight years. Prior to his work in .NET, Bob did commercial development with C++, DCOM, and ATL technologies. Prior to his career in computer science, Bob was a college physics professor and researcher. He holds a Ph.D. in physics and master's degrees in physics and engineering from the University of Wisconsin—Madison.

Acknowledgments

ALTHOUGH THIS BOOK HAS FOUR AUTHORS’ names on the cover, there are a number of people who have contributed in different ways to making this book better than it ever could have been without them.

We would all like to thank the Apress staff—they have been extremely supportive. This includes (but is not limited to) Gary Cornell, Dan Appleman, Stephanie Rodriguez, Grace Wong, Ami Knox, and anyone else at Apress who has been involved with this book—you turned our writing into a far better product than what we could’ve done on our own. We also thank the reviewers (Brock Allen, Robert Knudson, Chris Sells, and Nathan Smith) for their helpful comments and suggestions.

Jason would like to thank his wife Liz for her love and support (you’re the best) and all of his friends and family for just being there.

Tim would like to dedicate his contributions to Theresa, Ali, and Kate. They are the greatest!

Pete would like to first thank his wife, Kris, and his son, Ethan, for all their support during the writing process. He would like to thank and give all the glory to God, since he’s the one who has made all of this possible. Pete would also like to thank everyone at Magenic Technologies who has provided him the opportunities over the years to get him where he is now. He would like to thank Steve Waldner for being his mentor in this industry when he first got started. Lastly, he would like to thank Jason Bock for affording him the opportunity to write for this book.

Nathan would like to thank the members of the eternal bench crew, PZ and EL, for providing him with things to laugh about, things to gripe about, and excellent Thai food.

Introduction

"We just lost all of our JPEGs on our Web server,
and . . . um . . . we don't have a backup."

"Attention! If you have received an e-mail from Bob, do not open it."

"Could you look at this attachment? I think it's a virus. . . . "

VIRUSES. MALICIOUS E-MAIL ATTACHMENTS. Denial of service attacks. You can probably think of a number of other incidents that have happened to either you or a friend of yours on the job where a piece of unwanted code wreaked havoc on unsuspecting users. We've seen our share at the places we've worked at. In fact, all three of the quotes are from our jobs. The first incident happened when an employee opened an e-mail that contained a virus. Since he had the Web server mapped as a network drive, the images located on the server were destroyed. The second occurred when a consultant had the e-mail preview option on in Outlook and a virus was accidentally started. The company panicked, and ended up broadcasting a warning message over the intercom system. The last one happened when someone within the company triggered a virus, and management wanted one of us (Jason) to examine the attachment, as it looked like VBScript. They were hoping that they'd have a chance at understanding what kind of damage was being done to their systems.

We'd all like code to do what we want it to do. We'd all like to be able to open an attachment that appears to be an image without it e-mailing questionable Web page links to our friends and coworkers. But up until .NET, it's been rather difficult for developers to program security effectively. It's not impossible, but it's not as easy as using C++ to open a file either. Windows has always been accommodating to the user in terms of ease of use when it came to their applications. That, however, has lead to numerous security breaches and malicious executables doing their work on machines, sometimes unbeknownst to their users. In a nutshell, this has been pretty frustrating for both users and developers.

With .NET, however, Microsoft has made a concerted effort to make writing secure code a much easier endeavor that what it was. This new architecture also has the added benefit of making it easier to configure what code can and cannot do. Because the security-related classes are straightforward to use, this will help in ensuring that a corporation's machines and networks are virus free. At the same time, since .NET is a whole new ball game to everyone involved (including yours truly), it takes some time to become familiar with the classes to use them effectively. This book is an attempt to help to facilitate that learning process so you can get up to speed on .NET security programming.

Target Audience

This book is targeting the intermediate .NET developer who wants to understand how security works in .NET. Although the language of choice within this book is C#, the concepts are .NET-general and are not specific to any .NET language. A VB .NET or JScript .NET developer should be able to apply the concepts to their preferred .NET language with relative ease. We're also assuming that you know the fundamentals of .NET (for example, what an assembly is, what the difference is between a static and an instance method, and so on).

Source Code

We have created a number of small applications that we mention throughout the book. You can download the code from Apress's Web site, at `http://www.apress.com`. We have made every attempt to ensure that the code compiles and behaves as expected, but mistakes can occur. If you find a bug with the source code, or you find an erroneous statement in the book itself, please contact us at `jason@jasonbock.net`, and we'll make sure that updates are made accordingly.

Brief Contents

Contents

CHAPTER 1

The Basics of Cryptography and Security

In this chapter, you'll learn the fundamental concepts of cryptography and security. I'll define what cryptography is and what it is used for, and cover the basics of ciphers and keys and how they work. I'll demonstrate the difference between symmetric and asymmetric cipher algorithms and show you how they can be used in concert for creating signatures and certificates. Finally, I'll talk about cryptography and its relation to security in general.

The Essence of Cryptography

The definition of *cryptography* is pretty straightforward: it is the science of keeping messages secure. In essence, cryptography is the study of the mathematical algorithms and functions used to secure messages. These algorithms fall into two camps: restricted and open.

Restricted Algorithms

Restricted algorithms are those created by a person or an organization and are not available to the general public. For example, this could apply to a compiled COM server from TrustUs.com, which won't give you the source code nor give you any information as to how its algorithms work.

Open Algorithms

Open algorithms are published and are available to anyone for analysis. This could take the form of an algorithm found on VerifyUs.com, where you can download white papers that document the algorithm along with a compiled

Eiffel .NET Web service and its corresponding source code that encrypts and decrypts files for you based on the algorithm.

Experience has shown that you should always go with the open algorithm. Even if it turns out that an open algorithm is not as secure as a restricted one, you as a customer of the two algorithms have no way to verify that the restricted one is better (although if the open algorithm is weak, I'd consider finding another one). And, more often than not, restricted algorithms are hacked anyway by disassembling executables, so their "security by obscurity" methodology is ineffective at best. Determining if an algorithm can withstand attacks takes lengthy analysis and testing by many qualified professionals, so it's beneficial to use an open algorithm. You can easily find out if someone has found a gaping hole in the algorithm or if it is secure.

Next, let's go over the basic terms you'll need to understand to get the most out of subsequent chapters.

Basic Terminology

This section discusses the following terms:

- Plaintext
- Ciphertext
- Hashes
- Keys
- Symmetric algorithms
- Asymmetric algorithms
- Comparison of key types
- Random number generation

I'll begin with plaintext and ciphertext.

Plaintext and Ciphertext

Plaintext describes the state of a message that can be easily read or used by anyone or anything. This can be a text file or an executable. To alter the file such that it cannot be easily read by anyone, you use an *encryption* algorithm to turn the plaintext into *ciphertext*. The encryption algorithm is simply a mathematical function that takes a message's value and from it computes another value (the ciphertext). The hope is that the resulting ciphertext cannot be converted back into the original value easily. To get the file back to a usable form, you use a *decryption* algorithm. Figure 1-1 illustrates this process.

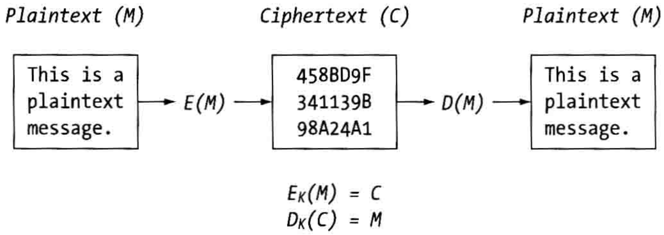


Figure 1-1. Encryption and decryption

The plaintext message is defined as M, and the ciphertext is C. In both situations, a function E (encryption) or D (decryption) operates on M or C, respectively, and produces the desired output.

A simple encryption algorithm that is widely known is Caesar's cipher. This takes each letter of the English alphabet and maps it to another letter. For example, one mapping would shift letters over a certain number of spaces. Therefore, a message such as "hello reader" could be changed to "ifmmp sfbefs," where each letter is shifted one letter in the cipher text.[1]

NOTE *Some books use the words* encipher *and* decipher *for* encryption *and* decryption, *respectively. I'll also use the word* ciphering *to describe the general process of changing a message from one form to another (this can be either the encryption or decryption process).*

Let's start looking at a number of encryption algorithms, beginning with hash algorithms.

[1] This algorithm is extremely easy to break, so I wouldn't suggest using it to encrypt sensitive corporate documents.