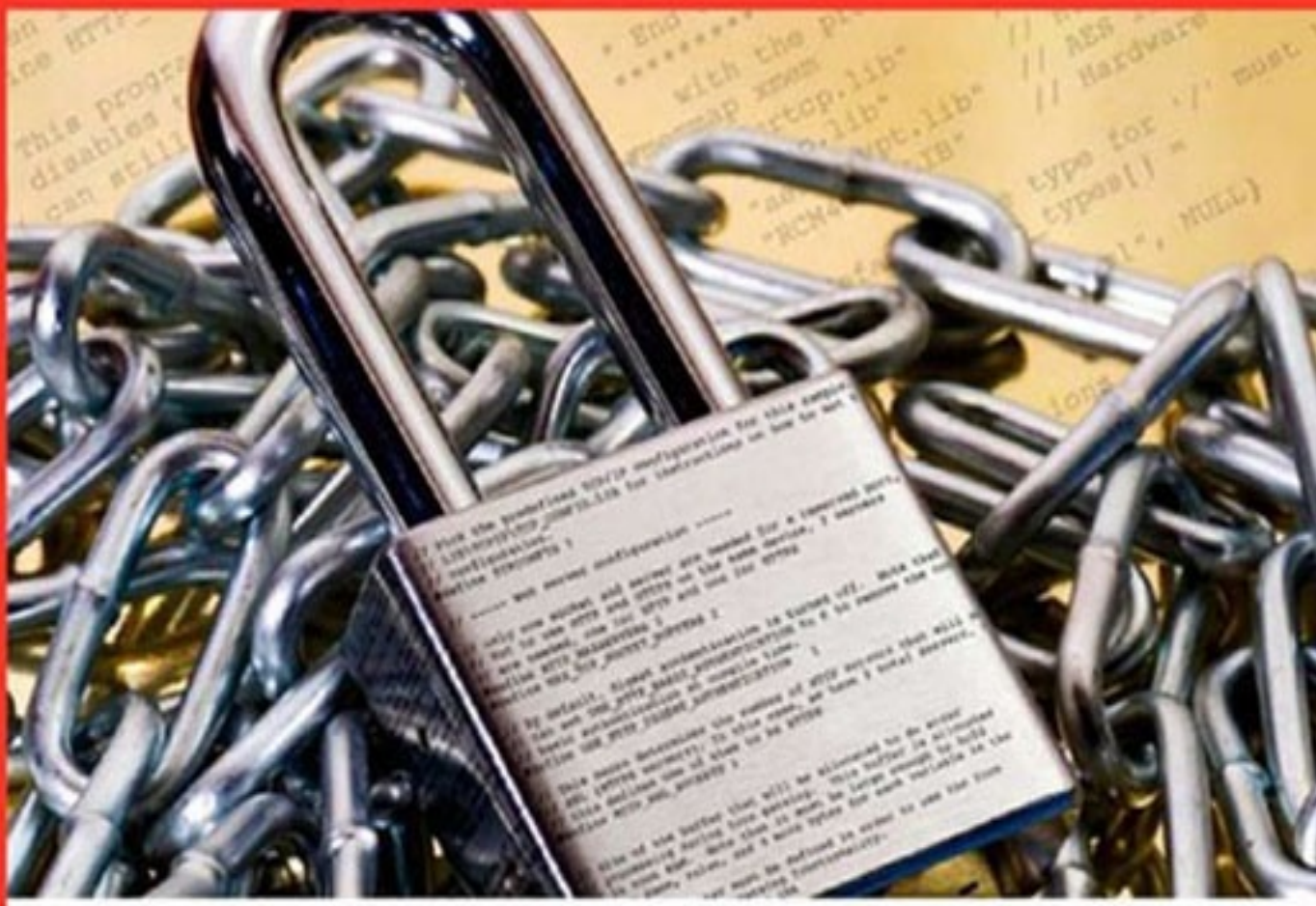
EMBEDDED TECHNOLOGY™ SERIES



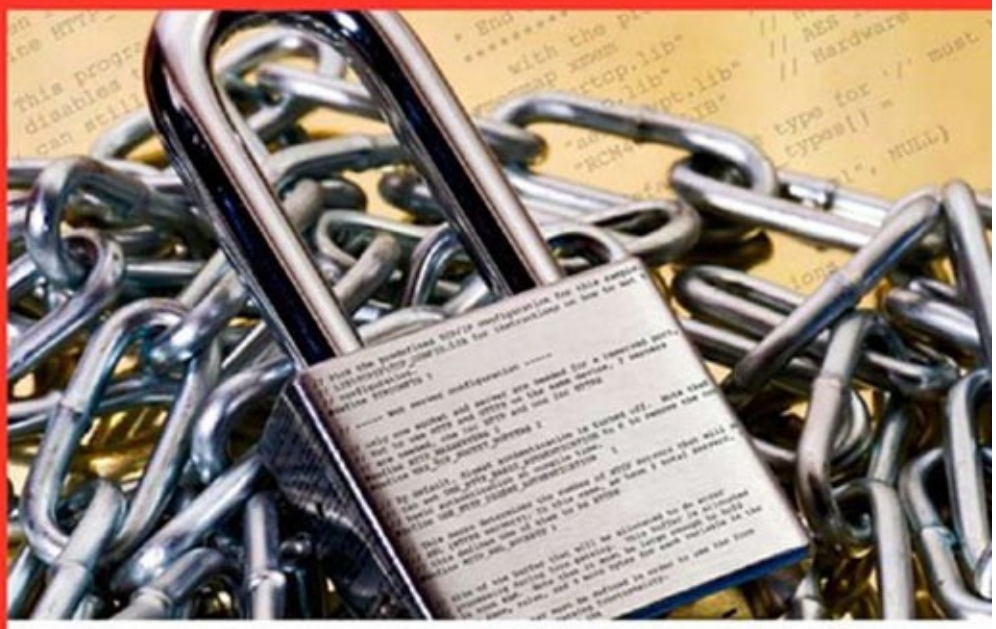
Practical Embedded Security

Building Secure Resource-Constrained Systems

Timothy Stapko

Newnes

EMBEDDED TECHNOLOGY™ SERIES



Practical Embedded Security

Building Secure Resource-Constrained Systems

Timothy Stapko

Newnes

Practical Embedded Security

Building Secure Resource-Constrained Systems

Timothy Stapko

Newnes

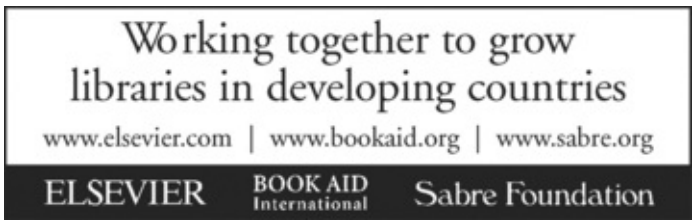
Table of Contents

Copyright

Newnes is an imprint of Elsevier
30 Corporate Drive, Suite 400, Burlington, MA 01803, USA
Linacre House, Jordan Hill, Oxford OX2 8DP, UK

Cover image by iStockphoto

Library of Congress Cataloging-in-Publication Data
Application submitted
British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.
ISBN: 978-0-7506-8215-2
For information on all Newnes publications
visit our Web site at www.books.elsevier.com
07 08 09 10 10 9 8 7 6 5 4 3 2 1
Printed in the United States of America



Preface

Living in a Connected World

1:37 AM. Hoover Dam, straddling the border of Nevada and Arizona, is quietly generating electricity for millions of Americans. The power plant, having recently been retrofitted with a new, remotely controlled automation system, is devoid of life, except for the blinking lights of the network hubs and automated hardware. Suddenly, the control room is ablaze with light, and the whirring of machines breaks the silence. The enormous floodgates open, a torrent of water rushing forth, sending a wave of destruction toward the unsuspecting communities downstream on the Colorado River. The turbines grind to a halt, plunging the desert into darkness. All the while, a teenager in Florida is laughing in the glow of his computer monitor.

Obviously, no one in his or her right mind would trust the control of Hoover Dam to a system with such gaping vulnerabilities, but the hyperbole of the example above does bring up an important point: as more and more of the world goes "online," we are putting more and more trust in the embedded systems that are designed to help us out. Obviously, something like the Hoover Dam would not be automated and connected to the Internet without a large investment in security, if it was automated at all. However, something far simpler, such as a home automation system, would likely not be subject to the same rigorous treatment as a vital hydroelectric power plant. This split between the security requirements of different embedded systems helps to illustrate the challenge of security design for embedded systems engineers. While the cutting edge of security is continually being pushed, low-end hardware and inexpensive systems are often left behind. However, these inexpensive systems are increasingly being networked and used to control more and more vital systems. This leads to an interesting and disturbing problem: Security implementations are often jealously guarded proprietary solutions that sell for thousands of dollars, which is directly at odds with the idea of using inexpensive microcontrollers. There are some options, such as various open-source implementations, but these can be unwieldy and are designed for PCs. If you want to design an inexpensive system and make it secure, there just are not many options.

One of the biggest problems with security in both the Hoover Dam example and home automation is the continual need for updates to keep up

with malicious hackers. Anyone with a PC running Microsoft Windows knows about this from the continual stream of updates and patches for various security issues. One way to alleviate the continual update problem is to design security into the system and develop a solid application to begin with. The primary goal of this book is to introduce the users of inexpensive microcontrollers and embedded processors to the basic practical application of security and to provide some tools and pointers to assist in designing more secure applications with limited resources.

Many of the topics discussed in this book are covered in depth in hundreds of academic papers and tomes filled with arcane symbols. If you are interested in the mathematical underpinnings of cryptography, you are going to want to look elsewhere. However, if you work with microcontrollers or inexpensive embedded systems and security has been something of interest but you have been intimidated by it, then this book is for you. Security is a hard problem, and a lot of very smart people have spent a lot of time working on it. The result is that the topic of security has taken on an intimidating air, especially when it comes to cryptography. This book aims to leverage the large body of work already done on security and adapt it for systems that usually aren't deemed powerful enough. As you will see, it is possible to implement security for some of even the most modest of architectures, such as porting AES to a PIC and using SSL on an 8-bit microprocessor (both of these are covered in extensive case studies of working implementations).

This book covers the practical side of implementing security for embedded systems, using publicly available and inexpensive proprietary implementations whenever possible. However, just having a cryptographic algorithm does not mean you have security. There are a number of issues to consider when using cryptography. We will cover some of them and hopefully provide some insight into how you can find them on your own.

Security in Shades of Gray

There is no such thing as perfect security. Think about it. As long as there is information that can be exploited, there will be someone trying to get it, and with enough resources, any security can be broken. The only way to be sure that information is completely "safe" is to destroy it and kill all the people who know about it, which obviously does no one any good. For this reason, secure systems are built using the idea of hazard tolerance—that is, each system has to have security that meets the requirements for the system. For example, a credit card system needs more security than your personal email (which we all know is completely *in*secure).[1] Unfortunately, security is an inherently difficult problem to solve, since the worst problems are necessarily