A Practical Guide to TPM 2.0

Using the Trusted Platform Module in the New Age of Security

Will Arthur
David Challener
With Kenneth Goldman

Apress open

A Practical Guide to TPM 2.0: Using the New Trusted Platform Module in the New Age of Security
Will Arthur & David Challener

ISBN-13 (pbk): 978-1-4302-6583-2

ISBN-13 (electronic): 978-1-4302-6584-9

Managing Director: Welmoed Spahr
Associate Publisher: Jeffrey Pepper
Lead Editors: Steve Weiss (Apress); Patrick Hauke (Intel)
Coordinating Editor: Melissa Maldonado
Cover Designer: Anna Ishchenko

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail `orders-ny@springer-sbm.com`, or visit `www.springeronline.com`.

For information on translations, please e-mail `rights@apress.com`, or visit `www.apress.com`.

About ApressOpen

What Is ApressOpen?

- ApressOpen is an open access book program that publishes high-quality technical and business information.
- ApressOpen eBooks are available for global, free, noncommercial use.
- ApressOpen eBooks are available in PDF, ePub, and Mobi formats.
- The user friendly ApressOpen free eBook license is presented on the copyright page of this book.

I dedicate my portions of this work to my wife Ruth, and sons Tim and Stephen — D. Challener

To pastor Jon MacKinney and Intel managers Linda Zavaleta and Jody Pfotenhauer, who encouraged me to pursue an engineering degree at an age when many men start thinking about retirement. To John Pennington and Monty Wiseman: for support and mentoring. To my wife, Tammy, and daughters, Casey, Megan, and Rachel: for your patience and support as I've ridden this high-tech roller coaster for the past 30 years. Most of all to Jesus Christ, my ultimate source of security. — Will Arthur

Contents at a Glance

Contents

About the Authors

Will Arthur is a senior staff firmware engineer in the Datacenter Engineering Group for Intel Corporation. He leads the development of authenticated code modules (ACMs) for the server version of Intel Trusted Execution Technology (TXT). As an active participant in the Trusted Computing Group's TPM and TSS working groups, he wrote the TCG TPM 2.0 System API and TPM 2.0 TAB and Resource Manager specifications, developed the TCG versions of the code that implements those specifications, and reviewed and edited the TPM 2.0 specification for readability and accuracy. Will has over 30 years of experience in low-level embedded firmware and software, the last 19 of those years with Intel. Will earned a BSCS in computer science from Arizona State University.

David Challener has been working on Trusted Computing since it started over a dozen years ago. He is currently co-chair of the TPM Working Group, and in the past has been chair of the TSS workgroup and on the TCG technical committee and Board of Directors. He has contributed to a number of other TCG specifications as well. He has a PhD in applied mathematics from the University of Illinois and currently works at The Johns Hopkins University Applied Physics Laboratory.

About the Technical Reviewers

Justin D. "Ozzie" Osborn is the chief scientist of the Commercial Device Operations Group at The Johns Hopkins University Applied Physics Laboratory. He has almost a decade of experience in software reverse engineering and embedded software development. He has worked on several projects that involved developing TPM software and performing vulnerability analyses of TPM solutions.

Monty Wiseman is a security architect in Intel's Data Center Group (DCG). His current projects include architecture for TCG, Intel's TXT technologies, Boot Guard, and other security initiatives. Monty has participated in and chaired the TCG PC Client working group and Security Evaluation working group for TPM 1.2. He participates in the TPM and other TCG workgroups and is Intel's representative on the TCG Technical Committee. Monty has 20 years of experience in desktop, network, and mainframe environments and has held security-related and other engineering positions at Novell, Fujitsu, and Control Data. He has been developing hardware and software for computers ranging from mainframes to microcomputers since 1975.

Acknowledgments

The authors gratefully acknowledge the contributions, edits, and suggestions from our external and internal reviewers:

- Ken Goldman wrote many of the chapters and ruthlessly reviewed the text for technical errors.
- Emily Ratliff and Jon Geater contributed their expertise and knowledge to the ARM and AMD sections of Chapter 22. Bill Futrall also contributed text to Chapter 22.
- Paul England, David Wooten, and Ari Singer helped us understand the specification.
- Paul England helped us understand Microsoft interfaces to the TPM.
- Monty Wiseman, Justin Osborn, Alex Eydelberg, Bill Futral, Jim Greene, and Lisa Raykowski did technical reviews.
- Patrick Hauke of Intel provided moral support and guidance throughout this process.
- We would also like to recognize the many direct and indirect contributions of the TSS and TPM WG members.

Introduction

"Seminal!"

"Riveting! I couldn't put it down until the last page."

"I'm exhausted from reading this book! It kept me up three nights in a row. Where's my Ambien when I need it?"

"The suspense was killing me. I just *had* to read it straight through!"

Although these responses to our book would be gratifying, it's doubtful that any book on digital security will ever garner this type of reaction. Digital security is the computer equivalent of disaster insurance. Few people care very much about it or give it much thought, and everyone hates paying for it … until a catastrophe hits. Then we are either really glad we had it or really sad that we didn't have enough of it or didn't have it at all.

We may sound like Chicken Little crying the "the sky is falling, the sky is falling," but mark our words: a digital security catastrophe is headed your way. We could quote a plethora of statistics about the rising occurrence of digital security threats, but you've probably heard them, and, quite frankly, you don't care, or at least you don't care enough. It's questionable whether any preaching on our part will make you care enough until you're personally impacted by such a calamity, but we'll try anyway.

When your reputation is tarnished, your finances are impacted, your identity is stolen, your physical well-being is threatened, your company's reputation and finances are harmed, and, quite possibly, your country is overthrown, then you'll wake up to the need for cyber security. But it might be too late then. Like people living in a flood zone, the question isn't whether the flood is coming, but rather when the disaster will hit and whether you'll be prepared for it. The time to buy digital-security flood insurance is now! Don't wait until the flood hits.

A Practical Guide to TPM 2.0 can be part of your digital-security insurance policy. The TPM was designed as one of the core building blocks for digital security solutions. The November 2013 "Report to the President: Immediate Opportunities for Strengthening the Nation's Cybersecurity" recommends "the universal adoption of the Trusted Platform Module (TPM), an industry-standard microchip designed to provide basic security-related functions, primarily involving encryption keys, including for phones and tablets. Computers and devices that incorporate a TPM are able to create cryptographic keys and encrypt them so they can be decrypted only by the TPM. A TPM provides this limited but fundamental set of capabilities that higher layers of cybersecurity can then leverage. Today, TPMs are present in many laptop and desktop personal computers. They're used by enterprises for tasks like secure disk encryption, but they have yet to be incorporated to any significant extent in smartphones, game consoles, televisions, in-car computer systems, and other computerized devices and industrial control systems. This needs to happen for such devices to be trustworthy constituents of the increasingly interconnected device ecosystem."

Our passion in writing this book is to empower and excite a rising generation of IT managers, security architects, systems programmers, application developers, and average users to use the TPM as the bedrock of increasingly sophisticated security solutions that will stem the rising tide of threats that are being aimed at us, our employers, and our civil institutions. Furthermore, the TPM is just plain cool. How many engineers, as children, played with simple cryptography for fun? The ability to send an encrypted message to a friend appeals to the secretive part of our human nature—the same part that enjoyed playing spy games when we were young. And besides being fun, there's something inherently, morally right about protecting people's assets from being stolen.

The TPM 2.0 technology can accomplish this. We believe in this technology and hope to make believers of you, our readers, as well. Our hope is that you'll get as excited about this technology as we are and "go out and do wonderful things" with it, to paraphrase Robert Noyce, one of Intel's founders.

Why a Book?

Technical specifications are typically poor user manuals, and TPM 2.0 is no exception. One reader of the specification claimed it was "security through incomprehensibility." Although the specification attempts to describe the functionality as clearly as possible, its prime objective is to describe how a TPM should work, not how it should be used. It's written for implementers of TPMs, not for application writers using TPMs.

Also, for better or for worse, the detailed operations of the TPM commands are specified in C source code. The structures are defined with various keywords and decorations that permit the Word document to be parsed into a C header file. Microsoft agreed with TCG that the source code in the specification would have an open source license and could be used to implement a TPM. However, although C can describe actions very precisely, even the best code isn't as readable as text. One of the major purposes of this book is to interpret the specification into language that is more understandable to average software developers, especially those who need to understand the low-level details of the specification.

Many readers don't need to understand the detailed operation of the TPM and just want to know how to use the various functions. These readers expect TSS (the TCG software stack) middleware to handle the low-level details. They're interested in how to use the new TPM features to accomplish innovative security functions. Thus, this book is just as concerned with describing how the TPM can be used as it is with explaining how it works. Throughout the book, as features are described, use cases for those features are interwoven. The use cases aren't complete—they describe what the TPM 2.0 specification writers were thinking about when those features were designed, but the specification is so rich that it should be possible to implement many things beyond these use cases.

Audience

In writing this book, we're trying to reach a broad audience of readers: low-level embedded system developers, driver developers, application developers, security architects, engineering managers, and even non-technical users of security applications. We hope to encourage the broadest possible adoption and use of TPMs.

Non-technical readers will want to focus on the introductory material, including the history of the TPM (Chapter 1), basic security concepts (Chapter 2), and existing applications that use TPMs (Chapter 4). Visionaries who know what they want to accomplish but aren't themselves programmers will also benefit from reading these chapters, because knowing the basic ways in which TPMs can be used may provide inspiration for new use cases.

Engineering managers, depending on their needs and technical expertise, can go as deep as they need to or want to. We hope that executives will read the book, see the possibilities provided by TPMs, and subsequently fund TPM-related projects. When they realize, for example, that it's possible for an IT organization to cryptographically identify all of its machines before allowing them onto a network, that true random number generators are available to help seed OSs' "get random number" functions, and that weaker passwords can be made stronger using the anti-dictionary-attack protections inherent in the TPM design, they may decide (and we hope they will) to make these features easily available to everyday people.

Security architects definitely need to understand the functions provided by TPM 2.0 and, depending on the applications being developed, dive deep into how the TPM works in order to understand the security guarantees provided. Linking disparate machines or different functions to provide trusted software and networks should be possible using TPM functionality as security architects get creative. Commercial availability of this capability is long overdue.

Application developers, both architects and implementers, are a significant focus of this book. These readers need to understand the TPM from a high-level viewpoint and will be especially interested in the use cases. TPM 2.0 is feature rich, and the use cases we describe will hopefully inspire creativity in developing and inventing security applications. Developers have to know the basics of symmetric and asymmetric keys and hashes in developing their applications—not the bit-by-bit computations, which are done in the TPM or support software—but rather the types of guarantees that can be obtained by using the TPM correctly.

We also want the book to be useful to embedded system developers, middle ware developers, and programmers integrating TCG technology into operating systems and boot code. The TPM now exposes more general-purpose cryptographic functions, which are useful when a crypto library isn't available due to either resource constraints or licensing issues. We hope that low-level developers will find that this book goes as deep as they need it to and that it serves as a critical tool in interpreting the specification. Toward this end, diagrams and working code examples are used to help clarify many concepts. We expect that embedded systems will increasingly use TPMs as the cost of the technology is reduced (making cryptographic computations cheap to integrate into embedded software) and as attacks on embedded software become more active.

Roadmap

If you're new to security or need a refresher, Chapter 2 gives an overview of the security concepts required to understand the book. This chapter provides high-level knowledge of cryptography: we explain symmetric and asymmetric keys, secure hash algorithms, and how a message authentication code (MAC) can be used as a symmetric key digital

signature. This chapter doesn't delve into the underlying math used to implement cryptographic algorithms; this isn't intended as a general-purpose security or cryptography textbook, because there is no need for most TPM 2.0 developers to possess that depth of knowledge.

Chapter 3 presents a high-level tutorial on TPM 2.0 and the design rationale behind it. It begins with applications and use cases enabled by TPM 1.2, all of which are also available in TPM 2.0, and then continues by describing the new capabilities that are available with the TPM 2.0 specification. This chapter should help you understand why people are excited about the technology and want to use it in their applications and environments.

Chapter 4 describes existing applications that use TPMs (currently, mostly 1.2). We assume that many of these applications will be ported to TPM 2.0. Some are open source, some are demonstration code written by academics to demonstrate what the TPM can do, some are applications that have been around a long time and that can be linked to use TPM features, and some are generally available applications written specifically to take advantage of the TPM's capabilities.

Chapter 5 provides a high-level orientation to the TPM 2.0 specification, offers pointers to critical parts of the specification, and explores some best practices for using the specification.

Chapter 6 describes the setup and use of the execution environments available for running TPM 2.0 code examples.

Chapter 7 discusses the trusted software stack (TSS). This is presented early in the book because succeeding code examples use various layers of the TSS.

Chapter 8 begins the deep dive into TPM 2.0 functionality with a description of TPM 2.0 entities: keys, data blobs, and NV indices.

Chapter 9 discusses hierarchies.

Chapter 10 covers keys.

Chapter 11 discusses NV indexes.

Chapter 12 explores PCRs and attestation.

Chapter 13 is one of the most in-depth chapters and is crucial if you're developing low-level code or architecting systems that make extensive use of sessions and authorizations.

Chapter 14 discusses enhanced authorization.

Chapter 15 explains key management.

Chapter 16 describes the TPM's auditing capabilities.

Chapter 17 examines decryption and encryption sessions and how to set them up.

Chapter 18 describes object, sequence, and session context management and the basic functionality of a resource manager.

Chapter 19 discusses TPM startup, initialization, and provisioning. In typical usage, these occur before keys and sessions are used, but knowledge of TPM entities and sessions is a prerequisite to understanding TPM initialization and provisioning. This is why we include this chapter after the previous three chapters.

Chapter 20 presents best practices for debugging TPM 2.0 applications.

Chapter 21 examines high-level applications that could use TPM 2.0 functionality.

Chapter 22 discusses platform-level security technologies that incorporate TPM 2.0 devices into their security solutions.

Assumptions

Although this is a technology book, we have tried to assume as little about our readers as possible. Code examples use C, and a working knowledge of C is useful. However, most of the concepts stand alone, and much of the book should be comprehensible to non-programmers. Security concepts are explained at a high level, and every attempt is made to make them understandable.

Some knowledge of the TPM 1.2 and 2.0 specifications is definitely beneficial but not required. We encourage you to download the TPM 2.0 specifications from www.trustedcomputinggroup.org so that you can refer to them as you read the book.