Foreword

The context of this book is the shared effort going back almost a decade to achieve standards for programmatic access to data on the Web. Dave Winer introduced the original concept of “RPC over HTTP via XML” in what we would now call a blog post in February 1998. From the outset, the aim has been to build a vendor-neutral, Internet-scale, software platform.

The Internet is a platform. A platform is made up of tools and runtimes. Internet tools run on all kinds of operating systems, as do the runtimes. The beauty of the net is its simplicity, its ubiquity and its lack of a controlling vendor. TCP is the runtime environment of the Internet. It’s deeply competitive with Microsoft, and it’s larger than Microsoft. The stronger TCP is, the more outside of Microsoft’s control it is, the more powerful everyone else will be. The Unix people have been in a struggle with everyone else over who owns the Internet. I’ve been writing about this for a long time. It would be great if they could let go of the struggle and realize that the Internet is owned by the universe, no single operating system or flavor of operating system can contain it. This is important because there is another layer coming on the Internet, a very simple one, that can build on COM, on Windows and elsewhere, and provide a flat playing field for everyone. It’s RPC over HTTP via XML. I believe it’s the next protocol for runtimes. (Dave Winer, http://davenet.smallpicture.com/1998/02/27/rpcOverHttpViaXml.html, February 1998)

Since 1998, the WS-* stack of specifications—including the core message format SOAP—has been developed at standards bodies such as W3C and OASIS. Moreover, robust implementations of these specifications are widely available, supporting probably all major programming languages and platforms. There have been criticisms, on grounds of the inefficiency of XML as a message format, and the combinatorial complexity of compositional specifications. Nonetheless, almost 10 years on, SOAP-based Web services are widely used, both within enterprises and on the public Web.

This book concerns security aspects of SOAP-based Web services. The core of SOAP security is a set of WS-* specifications—including WS-Trust and WS-SecureConversation—built on the security header defined by WS-Security. This core was introduced by IBM and Microsoft in a 2002 white paper. Again, from the outset the emphasis is on interoperability.

This document ... defines a comprehensive Web service security model that supports, integrates and unifies several popular security models, mechanisms, and technologies (including both symmetric and public key technologies) in a way that enables a variety of systems to securely interoperate in a platform- and language-neutral manner. (IBM and Microsoft, Security in a Web Services World: A Proposed Architecture and Roadmap, http://msdn2.microsoft.com/en-us/library/ms977312.aspx, April 2002)

Since this proposal there has been great progress in achieving secure, interoperable implementations of these security mechanisms. There are many implementations of WS-Security. Security is hard to get right, though, and there were vulnerabilities in some of the early specifications and implementations. Researchers found that
some classic man-in-the-middle and impersonation attacks were possible. For example, in the Samoa Project (http://research.microsoft.com/projects/samoa) we used formal methods to identify such vulnerabilities in early versions of some Microsoft implementations of WS-Security. Although much progress has been made, the formal verification of Web services security protocols and their implementations remains a challenging research problem. Still, pragmatic engineering guidelines now exist, at least for basic security—based on experience in practice, a recent specification (the WS-I Security Profile Version 1.0, March 2007) advises on how to achieve basic security guarantees when using WS-Security.

In the past 5 years, in parallel with the development of basic security standards for Web services, there has been a good deal of research on using and extending the basic security mechanisms, as well as investigating their benefits and limitations. This book is the first to present a selection of this Web services security research as a single collection.

The book includes critical experience reports of implementing WS-* security mechanisms on various platforms. It includes research on combining architectural styles such as model-driven design and grid computing with Web services. Several chapters report research on authorization policies and mechanisms for Web services, and in particular on the need for flexible, fine-grained delegation. The chapters cover a wide range of application areas of Web services including health information systems, financial services, and aspects of e-government such as taxation, virtual organizations in the chemical industry, and grid computing for science.

In keeping with the emphasis on interoperability and platform neutrality of the original Web service proposals, this book itself includes work using a variety of software platforms—including both Java and .NET—and development models—both open source and commercial.

I commend this book as an excellent overview of recent research on a variety of topics in Web services security.

Andrew D. Gordon, Microsoft Research