https://doi.org/10.48009/2_iis_2005_103-109

SECURITY AT THE EDGE: RETHINKING SECURITY IN LIGHT OF WEB SERVICES

Richard S. Swart, Utah State University, richard.swart@business.usu.edu
Bryan Marshall, Utah State University, bryan.marshall@usu.edu
Matthew E. Harris, Utah State University, mattharris@cc.usu.edu
Karen A. Forcht, Utah State University, karen.forcht@business.usu.edu
David Olsen, Utah State University, david.olsen@business.usu.edu

ABSTRACT

Web Services technology is gaining prominence in industry and is displacing established standards such as Electronic Data Interchange (EDI for B2B transactions). The highly distributed nature of Web Services, and the tight tie between the network and application layers, creates new vulnerabilities requiring Information Security professionals to re-evaluate their network and application security planning. This paper reviews Web Services technologies, challenges to their implementation, demonstrates nine ways that traditional assumptions regarding security are inapplicable to Web Services, and suggests possible solutions to these problems.

Keywords: Network Security, Web Services, Web Application Security, Security Architecture, IS technologies, Enterprise Resource Planning,

INTRODUCTION TO WEB SERVICES

Web Services are a group of related technologies built on open standards that allow computing systems to publish their services, discover other Web Services, and exchange data in an automated manner. Web Services appeal to corporations is the capability to describe their function, search out and dynamically interact with other Web Services via Web Services Description Language (WSDL), Universal Description, Discovery, and Integration (UDDI), and Simple Object Access Protocol (SOAP). According to Microsoft, Web Services provide a "loosely-coupled, language neutral, platform independent way of linking applications within organizations, across enterprises, and across the Internet" [5]. The power of the technology comes from its ability to alleviate the task of integrating multiple web applications, coordinating standards to pass data, protocols, and platforms [7].

It is highly significant when the entire industry adopts a new technology. Leading companies such as IBM, Oracle, HP, Novell, Microsoft, and Sun support Web Services and have integrated it into their products [6]. A number of major corporations have adopted Web Services, for example, Amazon, Continental Airlines, Dollar Rent-A-Car, Expedia, MSNBC.com, and NASDAQ.com [5].

The following figure, which is based on a figure in Scambray and Shema [7], demonstrates the publishing, discovery and binding features of these technologies.

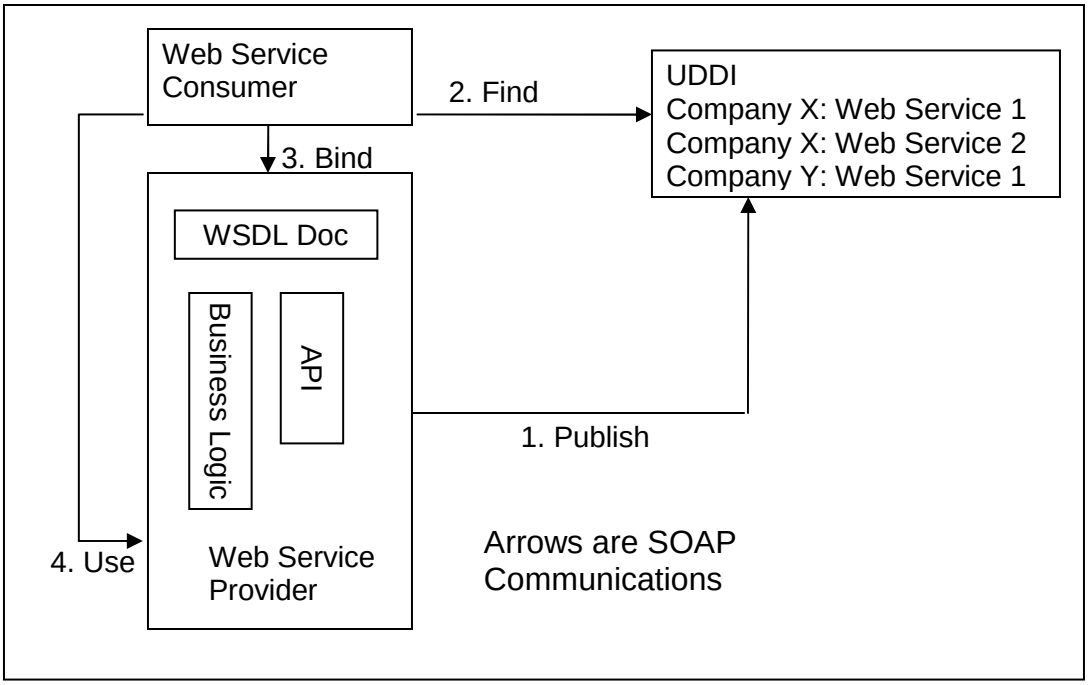

Figure 1. Interactions Between UDDI, WSDL, and Web Services over SOAP

WSDL documents describe the available Web Service and document the required message format, port number, and other parameters. Companies publish descriptions of their available Web Services and these are accessed using UDDI. Consumers generate SOAP messages, according to the defined protocol in the WSDL document. They include the action they would like to take according to what a producer has to offer. These actions are really remote method calls. It is important to recognize that unlike a web server delivering content, Web Services tie directly into the organization's enterprise systems and provide a direct connection to underlying Application Program Interfaces (APIs) and expose the organization's business logic.

INFORMATION SYSTEMS SECURITY

The value of information to an organization increases as the organization becomes more dependent on that information [1]. Web Services are essentially used in B2B commerce. They are used to automate the exchange of data between core business applications. Given that these services directly interact with Enterprise Resource systems, facilitate supply chain linkages, and act autonomously from human input; the need to maintain the key security services of confidentiality, integrity, and availability of data is crucial. It is well-known that many security breaches go undetected [11]. A lack of managerial awareness is a major concern for organizations as they plan to secure their data [2]. Few security managers are trained in Web Services and as a result many organizations either resist adopting Web Services due to security concerns, or inappropriately rely on traditional security mechanisms that do not work when using Web Services. Steinke suggests that when discussing the benefits of implementing information security with management, an information security professional should ask, "What is the cost of not recognizing and protecting against the threat of information system invasion [intrusion] to the firm?" [9].

The following section of the paper documents nine ways in which Web Services violates traditional assumptions about security and can expose corporate data to invasion if not properly addressed. Each point includes a brief description of a control that addresses these vulnerabilities.

RETHINKING SECURITY FOR WEB SERVICES

Though there does not appear to be any research documenting the core assumptions that inform security planning, the authors reviewed security literature and found nine common assumptions that Web Services invalidate or challenge. The following table briefly describes our findings.

Table 1. Traditional Security Assumptions Violated by Web Services and Possible Solutions

Traditional Security Assumptions	**Web Services Perspective**	**Solutions**
DMZs and application level gateways protect applications	Web Services are tunneled over HTTP and SSL and thus pass-through firewalls	Read SOAP messages, then filter on content–Segregate Web Services from the network
Transport level security can manage trust	The multiple hops in Web Services loses the security context of SSL	Apply encryption to elements within the SOAP message or attach a digital signature to the SOAP message request
Schemas are a formatting issue	Schemas are a data validation tool and their manipulation can crash services	Block external references in XML documents and SOAP requests - Protect schemas with an intrusion detection system
Authentication applies to users	Authentication needs to apply to machines, WSDL depositories, and XML schemas	Use SAML to embed the authentication status and permissions in messages
Security can be separated from applications	In Web Services, the application and network layer security issues are merged	Scan and capture XML and SOAP at the edge of the network to verify the contents
Security through obscurity -- web application functions should be disguised	The nature of WSDL provides verbose and explicit information about the services, and can expose server architecture and OS details	Use application logic or an XML/Web Services appliance to block SOAP messages that are not from a trusted source
Broad implementation improves security	Implementation of Web Services can expose underlying business logic and core business applications to new attacks	Monitor data flows and Web Services behavior in addition to packet flows
Human monitoring is mandatory and effective	Monitor at the data level and at the periphery of a network since systems are fully autonomous	Examine content specific Web Services usage for anomalies
Web applications are not core to business processes	Web Services enable automated B2B transactions of core business processes	Audit Web Services developers Verify Web Services stubs do not expose vulnerabilities

Assumption 1. DMZs and Application Level Gateways Protect Applications

Since SOAP and application data is tunneled through ports 80 and 443, Web Services are able to pass though firewalls. A network-centric firewall only checks to see if the transport HTTP is valid. It does not check whether the SOAP message is malicious, which could crash the application. Unlike a traditional Demilitarized Zone (DMZ) that contains only a web server, a Web Services server provides a direct link into the business's core applications. Layering Web Services on top of legacy applications exposes the underlying business logic vulnerabilities.

Traditional firewalls do not parse SOAP or eXtensive Markup Language (XML) messages and are unable to secure environments using Web Services. Packet filters cannot recognize the behavior of Web Services consumers. It is no longer enough to rely on application layer firewalls. Web Services providers need a system that reads deep into the SOAP message and enforces policies [8]. Since web services often need to initiate outbound communication with the Internet, it is often a good idea to segregate networks running Web Services from those providing web servers that only provide content [4].

Assumption 2. Transport Level Security Can Manage Trust

In transport-level security, Secure Socket Layer (SSL) and *username: password* pairs are used per application session. However, SOAP is a connectionless protocol, and connection oriented techniques such as SSL are insufficient to ensure confidentiality since they only apply to end-to-end communication. SOAP messages move across multiple legs of a communication link, and when this occurs, transport centric authentication, like SSL, cannot assure data integrity and confidentiality across multiple hops. The security context is essentially lost. SOAP messages can be sent in any transport method. This requires that providers and consumers apply security to the SOAP message itself, rather than relying on the transport over which it is sent. This is crucial to any meaningful access control scheme. Though SSL only protects the first hop of a communication, it is still recommended that SOAP messages be transported with the Secure Hypertext Transfer Protocol (HTTPS) to provide one layer of confidentiality. Trust related challenges are addressed by applying encryption to elements within the SOAP message, or by signing and attaching a digital signature to the SOAP message request.

Assumption 3. Schemas Are a Formatting Issue

WSDL relies on XML schemas to define data types and format the messages. Traditionally, XML schemas are seen as a formatting device; however, schemas describe pre-processing instructions. If a schema is modified (known as schema poisoning) the system can be crashed by an attacker. Given the interdependence of systems in B2B networks, if SOAP syntax is not checked before the message touches the business logic, the improper syntax can affect multiple systems. One of the key controls for this attack is to block external references within XML documents and SOAP requests. This prevents schema poisoning from attacks on third party schemas. The organization's schemas should be protected with a system such as Tripwire (www.tripwire.com) that immediately notifies the network administrator of any change to an identified file.

Assumption 4. Authentication Applies To Users

Since most authentications within a client/server model occur within the user-interface logic, this type of security is no longer relevant with Web Services. In today's web, software agents and computer system interact with each other autonomously. As the role of Service Oriented Architecture continues to expand, identity will move beyond the user towards authenticating the identities of the services themselves along with the identity of the host servers [3]. Corporations should allow only trusted consumers to read, write or alter data. Web service operations need security-related restrictions on invocation by only trust consumers.

Corporations should use the Security Access Markup Language (SAML) that defines an XML vocabulary for sharing security "assertions" such as authentication and authorization assertions, and provides for management features. It includes a protocol for binding to SOAP resources and for conveying requests for authorization or authentication to applications. It also provides for the association of digital signatures to assertions, such as: "Alice asserts that Bob has clearance to perform a certain action." A series of elements are defined and attributes such as "Permit", "Deny" or "Intermediate" are specified. Users should embed authentication status and permissions with a particular message using SAML. It should be noted that despite this solution, many adopters of Web Services adopt the communication features of Web Services without utilizing SAML (W. Swenson, personal communication, February 22, 2005).

Assumption 5. Security Can Be Separated From the Application

Many vendors of applications, such as Database Management Systems and Enterprise Resource Planning packages, have relied on the network providers such as CISCO and Bay Networks to provide security. Implicit in this assumption is the belief that application security can be a separate issue from network security. However, Web Services essentially merge the network and application layer and security must be applied to the Web Services application. Corporations should scan and capture XML and SOAP at the edge of a network and perform content sensitive data checks before the data reaches the core applications. At this level, messages should be parsed for inappropriate content such as SQL statements. The more that security can be moved to the edge of a network providing Web Services, the corporation will incur less costs for fixing application level vulnerabilities.

Assumption 6. Security through Obscurity

Since WSDL documents contain explicit instructions on how to communicate with previously private applications, they can cause a serious security breach if the Web Services are compromised. Since producers want consumers to know how to send proper messages, most Web Services error messages are detailed and let the consumer know as much as possible about failures and exceptions. Thus, an attacker can study the responses to learn about the system. Whether or not the operations have validation code and the system's responses to invalid data can provide details about the parser and code logic. A hacker can develop multiple attack patterns using the expressiveness of SOAP messages. Given that a WSDL document provides information on the web application's purpose, functional breakdown, entry points, message types, and that it contains an XML Schema, an attacker can create valid requests. The hacker

then uses probing attacks – scanning WSDL documents and sending different message request patterns until a breach is identified. One significant problem with global enterprise deployments is having undocumented or legacy operations still up and running as Web Services. A solution would be to create application logic or use an XML/Web Services firewall appliance to block SOAP messages attempting to bind to a Web Service unless the request comes from a trusted source.

Assumption 7. Broad Implementation Improves Security

The traditional assumption is that the maturing and increasing implementation of a technology leads to improved security. However, as more XML parsers, SOAP processors and WSDL applications are being used, additional vulnerabilities are being uncovered. Web Services is essentially a new technology and there are many unknown vulnerabilities when one considers the interaction of XML parsers, schema repositories, WSDL documents, client/server conversations, and SOAP stack protocols such as UDDI. New forms of attack at the SOAP/XML data level target weaknesses in Web Services programming, technology, and architecture. An example is the external reference attack: SOAP documents can build themselves by referencing URLs that point to third party content. If this content is malicious, the SOAP remote procedure call opens TCP or arbitrary files that expose the core business application to malicious code. It is also possible to route SOAP message though intermediaries. If the XML tags containing these routing instructions are compromised, the SOAP messages can be routed to a non-trusted location. Immunization against these attacks involves enhancing business behavior analysis. Businesses must move beyond analyzing packet flows to the monitoring of data flows and Web Services behavior.

Assumption 8. Human Monitoring Is Mandatory and Effective

The goal of Web Services is the full automation of business systems. In such an environment, human verification has no place. On demand virus scans, network checks and balances, and workflow approvals are ineffective in a fully automated system. An attacker can create well-formed SOAP messages as specified in the WSDL document that will avoid detection by firewalls and parsing checkers. Based on this vulnerability, validation needs to occur at the data level. Despite these limitations, Web Services providers and consumers still need to be able to know whether to trust information, but also need to know whether there is threat-related activity associated with the information being received. The only effective way to monitor Web Services is to look at content specific Web Service usage for anomalies [8]. Web Services security uses a positive model. It monitors applications to ensure that they behave as intended, without specific reliance on attack signatures [10].

Assumption 9. Web Applications Are Not Core to Business Processes

When the World Wide Web became prominent, many corporations viewed the web as an adjunct to their marketing or sales effort and not a part of their core processes. Today, Web Services are now a primary business and information technology consideration due to the amount of sensitive information being exchanged via the Internet. Any legacy application or architecture can be extended with a Web Services responder stub. This acts as a go-between for the application and

transmits the data received from the application in a standard message response over SOAP. The result of extending applications with Web Service stubs is to expose mission critical systems to security attacks over public internet connections. The possibility of severe loss from security breaches is magnified since Web Services penetrate deep into business logic and into back-office systems. Careful corporations should also audit Web Services developers since insider attacks can be extremely costly when one considers that they are accessing critical systems. Most legacy systems contain no business logic for detecting hacking. System developers need to ensure that the Web Service stubs do not expose existing security vulnerabilities.

CONCLUSION

Web Services will continue to be adopted by major companies to facilitate autonomous B2B data exchange and transactions. This fully automated and autonomous model requires that security be applied at the edge of a network through the use of specific Web Services, aware firewall appliances and through the monitoring of Web Services behavior. Shifting from a packet focused security model to a data-level model challenges many traditional assumptions of security planning, but is necessary to maintain a secure environment when utilizing the strategic advantages available with Web Services. With effective monitoring and implementation of the authentication of business partners, it is possible to use Web Services while maintaining an adequate security posture.

REFERENCES

1. Forcht, K. A. & Ayers, W. C. (2001). Developing a Computer Security Policy for Organizational Use and Implementation. *Journal of Computer Information Systems, 41*(2), 52-57.
2. Lundgren, T. (1994). User System Security. *Journal of Computer Information Systems, 34*(3), 32-40.
3. McAllister, N. (2004, September 6, 2004). SOA and Identity: Joined at the Hip. *InfoWorld*, 42.
4. McClure, S., Scambray, J. & Kurtz, G. (2003). *Hacking Exposed: Network Security Secrets and Solutions* (4th ed.). New York: McGraw-Hill.
5. Microsoft. (2005). *Web Services Case Studies*. Retrieved March 5, 2005, from http://www.microsoft.com/net/business/casestudies.asp
6. O'Neil, M. (2003). *Web Services Security*. New York: McGraw-Hill.
7. Scambray, J. & Shema, M. (2002). *Hacking Exposed Web Applications*. New York: McGraw-Hill.
8. Schultz, K. (2004, September 6, 2004). XWall Fends Off Web Services Attacks. *InfoWorld*, 30-31.
9. Steinke, G. (1998). A Task-Based Approach to Implementing Computer Security. *Journal of Computer Information Systems, 38*(1), 47-54.
10. Violino, B. (2004, May 2004). Decoding Application Security. *CSO, 3*, 48-53.
11. Whitman, M. (2003). Enemy at the Gate: Threats to Information Security. *Communication of the ACM, 46*(8), 91-95.