Web Services: Security

Lynda L. McGhie
Information Security Officer (ISO)/Risk Manager, Private Client Services (PCS), Wells Fargo Bank, Cameron Park, California, U.S.A.

Abstract
This entry discusses a core set of security functions that must be addressed in any successful security infrastructure. Web services security is introduced, defined, and discussed within the framework of the technology and tools that are already in place within a particular environment and how one can use the security control capabilities within Web services technologies to provide similar functionality.

INTRODUCTION
IT security professionals are challenged to keep abreast of constantly evolving and changing technology and, thus, new and complex security solutions. Often, it is impossible to implement new security control mechanisms concurrently with the implementation of new technology. One challenge most often facing Information Systems Security Organizations (ISSOs) is the competition with other business and IT departments for a share of IT budgets. Another is the availability of resources to include trained security architects, engineers, and administrators. In many large and complex organizations, the IT organization and hence the security support functions are often fragmented and spread throughout the enterprise to include the lines of business. This is a good thing because it increases awareness and builds support for the untenable task at hand, yet it most often results in the ongoing implementation of a very fragmented security infrastructure and company security posture.

Security is typically not brought into the beginning of any project, application, or systems development lifecycle. More often, security is asked to sign off just prior to implementation. How then does the ISSO catch up with or stay abreast of the constantly changing IT and business environment while ensuring that the enterprise is secure and security support services are optimized and effective? This entry looks at that challenge with regard to Web services and suggests a roadmap or a blueprint for integrating Web services security into an existing enterprise security strategy, policies, architecture, and access management function. A primary goal is to ensure that the above support components are designed to smoothly integrate new technology and applications without a great demand on resources or disruption. Another goal is to optimize previous and existing investments, yet be able to smoothly integrate new solutions.

Web services introduces a whole new set of standards, capabilities, vocabulary, and acronyms to learn and relate back to existing threats, vulnerabilities, and security solutions. The entry discusses a core set of security functions that must be addressed in any successful security infrastructure. Web services security is introduced, defined, and discussed within the framework of the technology and tools that are already in place within a particular environment and how one can use the security control capabilities within Web services technologies to provide similar functionality.

It is hoped that by framing legacy functionality and its associated toolset in light of introducing a new technology, standards, and functionality, the discussion will have a solid baseline and point of reference, resulting in greater understanding and utility.

This entry focuses on Web security services standards—what they are and what they do. Security should be applied only when and to the extent required, and the security architecture design should be as simplistic as possible and require as few resources to maintain as possible. To the extent feasible, access controls should be based on group-level policies, and individual access rules should be the exception rather than the norm. Remember that baseline security policies and access control requirements should originate from company business requirements and corporate threat profiles, not from technology. In this case, technology is not the driver. Sure, security tools are evolving fast and furiously, and for those of us who have been in security for some time, we finally have the wherewithal to actually do our jobs, but we need to stay in check and not over-design a Web services security solution that over-delivers on the baseline requirements.

This entry concludes with a discussion of changes to firewalls and traditional external perimeter controls, as well as Web services threat models. It also looks at the evolutionary aspects of the legal framework now so intrinsic to any enterprise security program.
Web services security introduces a whole new set of security capabilities and functionality. Web services have been slow to take off and evolve. Standards have existed for several years and have really matured, and for the most part vendors are aligned and in agreement. There are a few vendor alliances and a minimal number of groups with differing approaches, although more or less in agreement. This is different from what was seen in the past when other service-oriented infrastructures were proposed (e.g., CORBA and DCE). This alone will enhance the potential for success with Web services standards. Companies have been slow to move toward embracing Web services for various reasons: up-front investments, the newness of the technology, and also the maturity of the security solutions. Nowadays, companies are moving from point-to-point or service-to-service internal applications to enterprise-wide and externally facing, many-to-many implementations.

When the World Wide Web (WWW) was first introduced, it was viewed more as an Internet tool and certainly not as a production-worthy system within the enterprise. First uses included internal reporting, where data was transported from legacy applications to the Web environment for reporting. A later use was in browser GUI front-end-to-legacy applications. Still later as security became more robust and layered or defense-in-depth security architectures enabled the acceptance of greater risk within the Internet and Web application environments, Web-based applications began to move to DMZs (protected networks between the internal corporate network and the Internet). Eventually, these applications moved out to the Internet itself. Today e-business applications are served from customer-facing portals on the Internet, and many companies conduct their entire business this way, communicating with partners, supply chains, and customers.

With Web services, this evolution will continue and become more cost-effective because application development will become easier, more standardized, and the time to market for applications will greatly decrease. Along with this will come reusable services and functionality and a more robust set of capabilities than has ever been seen before in the application space. However, the road to Web services security will be a scary ride for the ISSO team.

In further examining the capabilities and solutions for Web services security, remember that the same vulnerabilities exist. The exploits may take a slightly different path, but the overall security solutions and functions do not change—that is, threat and vulnerability management, alert and patch management, and crisis management. Keep in mind some of the same baseline security tenets in going forward, including protecting data as close to the data as possible. Where possible, use the native capabilities within the operating system or vendor product, and strive to use a dedicated security product as opposed to building individual security solutions and control mechanisms into each application. There are differing approaches to doing this today within Web services, and this entry examines some of the choices going forward.

As Web services security standards continue to coalesce, vendors align, products evolve, and vendors either merge, get bought out, or fall by the wayside, the number of directions, solutions, and decisions decreases. But that does not change the complexity of the problem, or get us any closer to the right solution set for each company’s unique set of today’s requirements. How each company solves this problem will be unique to its business vertical, customer, and stakeholder demands, existing IT infrastructures and investments, resource availability, and business posture and demand.

One needs to choose from the resultant set of vendors and decide on looking at suites of products and functionality from a single vendor (Microsoft, BEA Systems, IBM, etc.) or adding third-party vendors to the mix, such as Netegrity, Sanctum, and Westbridge. ISSOs will traditionally approach this dilemma by reducing the number of products to support and administer separately. They will be looking for front-end provisioning systems and back-end integrated and correlated audit systems. They will also strive to reduce some of the security products, hoping that vendors combine and add functionality such as network firewalls, moving to incorporate application layer functionality. However, in the Web services security space, there is a need for new products because the functionality one is trying to secure is new, and existing products do not address these problems or have the capability to secure them.

Okay, there is a new technology, and for once there is agreement on a set of standards and solutions and therefore fewer choices to make and vendors to select, but how does one decide? If there is a heavy investment in one vendor and that vendor is in one or more alliances, it makes sense to join up there. If one is an agnostic or has some of everything, the decision becomes more difficult. This author suggests that you inventory your legacy, document your direction, and conduct a study. Look at a business impact analysis based on where integrated business processes are going at your company in the future. Which applications will be invested in, and which will be sun-setting?

PROFITING FROM PREVIOUS SECURITY INVESTMENTS

Current security investments, particularly at the infrastructure layer, are still necessary, and enhancements there should continue with the goal of integrating to a common, standard, and single architecture.
The same components of a well-planned and well-executed security implementation need to remain and be enhanced to support Web services. Unfortunately, as Web services standards continue to evolve, as applications migrate to Web services, and as vendors and partners adopt differing standards, approaches, and directions, the ISSO’s job gets more difficult and more complex. There will be some false starts and undoubtedly some thrown away, but nevertheless it is best to get an early start on understanding the technology and how it will be implemented and utilized in a particular environment. And finally, how it will be integrated and secured in your environment. Most likely, one will need to support a phased Web services security implementation in tools and capabilities become available and integrate. One might be balancing and straddling two or more security solution environments simultaneously, while keeping in mind the migration path to interface and eventually integrate to a single solution.

Investments in security infrastructure are still of value as a baseline framework and a springboard to Web services security. Also, look to augmentation through people, process, and other technology to determine what to keep, what to throw away, and what to adapt to the new and emerging environment. Do not count on having fewer security products or capabilities in the future, but certainly do count on automating a lot of today’s manual processes.

Looking then to understanding the new through the old, we now consider and address the basic components and security imperatives embodied in a typical security model:

- **Confidentiality**: Data or information is not made available or disclosed to unauthorized persons or processes.
- **Integrity**: The assurance that data or information has not been altered or destroyed in an unauthorized manner.
- **Availability**: Data or information is accessible and useable upon demand by an authorized person.
- **Authentication**: The verification of credentials presented by an individual or process in order to determine identity.
- **Authorization**: To grant an individual permission to do something or be somewhere.
- **Audit**: Collects information about security operating requests and the outcome of those requests for the purposes of reporting, proof of compliance, non-repudiation, etc.

Table 1 compares today’s Web security tools, standards, and capabilities to the new Web service security capabilities with respect to the model.

In migrating a security toolset, one will be using many of these control mechanisms together, and hopefully as one’s company becomes more standardized to Web services, one will leave some of these behind. Nevertheless, existing investments are salvageable and still need to be augmented with people, processes, and technology, as well as a combination of technical, physical, and administrative controls.

### WEB SERVICES APPLICATIONS

A Web services application is an application that interacts with the world using XML for data definition, WDSL for service definition, and SOAP for communication with other software. Web services application components operate across a distributed environment spread across multiple host systems. They interact via SOAP and XML. Other services include UDDI-based discovery (Web services directory) and SAML-based federated security policies.

### WEB SERVICES

- A stack of emerging standards that define protocols and create a loosely coupled framework for
programmatic communication among disparate systems (The Stencil Group).
- An emerging architectural model for developing and deploying software applications (The Stencil Group).
- Self-contained, modular applications that can be described, published, located, and invoked over a network—generally, the World Wide Web (IBM).

**SERVICE-ORIENTED ARCHITECTURES**

Service-Oriented Architectures (SOA) is a development in distributed computing, wherein applications call other applications over a network. Functionality is published over the network, utilizing two distinct principles: the ability to find the functionality and the ability to connect to it. In Web services architecture, these activities correspond to three distinct roles: Web services provider, Web services requestor, and Web services broker.

SOA is a process and an architectural mindset that enables a type of IT structure to be put in place. It requires significant coordination and integration throughout the enterprise, to include IT and business organizations. SOA is a continuous process that changes the way IT technologies are developed and used. One of the benefits of SOA is that an organization does not have to change all of its applications right away to derive a benefit. Companies can pursue a strategy of making some of their applications services-oriented and gradually migrating future applications. Often, a significant ROI is attained at all levels. Because SOA is all about reuse, the first project often yields a positive ROI.

Fig. 1 defines and illustrates the interaction and interface of SOA layered components.

**SIMPLE OBJECT ACCESS PROTOCOL**

Simple Object Access Protocol (SOAP) provides the definition of XML-based information that can be used for exchanging structured and typed information between peers in a decentralized, distributed environment.

SOAP is fundamentally a stateless, one-way message exchange paradigm, but applications can create more complex interaction patterns (e.g., request/response, request/multiple responses, etc.) by combining such one-way exchanges with features provided by an underlying protocol or application-specific information. SOAP is silent on the semantics of any application-specific data it conveys, as it is on issues such as the routing of SOAP messages, reliable data transfer, firewall traversal, etc. However, SOAP provides the framework by which application-specific information can be conveyed in an extensible manner. Also, SOAP provides a full description of the required actions taken by a SOAP node on receiving a SOAP message.

A SOAP message is basically a one-way transmission between SOAP nodes—from a SOAP sender to a SOAP receiver—but SOAP messages are expected to be combined by applications to implement more complex interaction patterns, ranging from request/response to multiple, back-and-forth “conversational” exchanges.

**CONFIDENTIALITY**

When data is stored, access control or authorization can potentially suffice for protection; but when data is in transit, encryption is often the most appropriate way to ensure confidentiality. Remember that decisions regarding what technology to use and in what layer of the OSI stack to place security may or may not be a function of technology, but may be more associated with the business process being addressed and the sensitivity and criticality of the information processed. Secure Socket Layer (SSL) can be used if the SOAP request is bound to HTTP or IPSec at the network layer. XML encryption enables confidentiality across multiple SOAP messages.
and Web services. If SSL is used alone, there is a gap at each endpoint.

**Digital Signatures and Encryption**

Digital signatures perform a key role in Web services, including non-repudiation, authentication, and data integrity. The XML signature is a building block for many Web security services technologies.

This functionality has been provided previously for Web applications utilizing S/MIME and PKCS#7. Public key cryptography standards (PKCS) is a voluntary standard (created by RSA and others). The W3C Digital Signature Working Group (“DSig”) proposes a standard format for making digitally signed, machine-readable assertions about a particular information resource. Prior to XML signatures, PKCS could digitally sign an XML document, but not in a standardized DML format. It was also not possible to sign just a portion of a document. Binding a signature to a document already existed for e-mail using S/SMIME, therefore enabling the recipient to verify the integrity and non-repudiation of the signer.

**Authentication and Authorization**

Secure Assertion Markup Language (SAML) defines a framework for exchanging security information between online business partners. More precisely, SAML defines a common XML framework for exchanging security assertions between entities. SAML’s purpose is to define, enhance, and maintain a standard XML-based framework for creating and exchanging authentication and authorization information. SAML is different from other security systems, due to its approach of expressing assertions about a subject that other applications within a network can trust. These assertions support specific entities, whether or not those entities are individuals or computer systems. These entities must be identifiable within a specific security context, such as human who is a member of a workgroup or a computer that is part of a network domain. An assertion can be defined as a claim, statement, or declaration. This means that assertions can only be accepted as true subject to the integrity and authenticity of the entity making the assertion (entity making claim/assertion must have authority). If one can trust the authority making the assertions, the assertion can be accepted as true with the same level of certainty as any other certification authority can be trusted. Additionally, SAML defines a client/server protocol for exchanging XML message requests and responses.

SAML is concerned with access control for authenticated principals based on a set of policies (see Fig. 2). There are two actions that must be performed with respect to access control in any enterprise system: 1) making decisions about access control based on a set of policies, and 2) enforcing those decisions at the system level; SAML provides two functions: policy decision point and policy enforcement point.

SAML is critical to the ability to deliver Web services applications because it provides the basis for interoperable authentication and authorization among disparate systems, and it supports complex workflows and new business models. The adoption of SAML by vendors of operating systems, identity and access management systems, portals, and application servers will simplify security integration across heterogeneous environments (Gartner IGG-05282003-02).

**Extensible Access Control Markup Language**

Extensible Access Control Markup Language (XACML) is being produced by the OASIS standards body to define an XML vocabulary to express the rules on which access control decisions are based. XACML enables interoperability across differing formats, enabling single sign-on, etc. XACML defines both architecture and syntax. The syntax is a means of defining how various entities process these XACML documents to perform access control:

- Defines rules to allow access to resources (read, write, execute, etc.) (more granular, defines XML vocabulary)
- Defines the format of the rules (rules for making rules) (policies)
- Policy exchange format between parties using different authorization rules (interoperability across disparate formats for SSO)
• Access control: ACLs and RBACs = syntax and architecture
• Authentication, confidentiality, integrity, and privacy

Focus on deploying Web services security and management infrastructures, as opposed to building application-based security. Much of Web services security can be implemented external to the application. Enterprises should plan to deploy a Web services management system or a security infrastructure that remains centralized, that is available for distributed Web services applications, and that is managed outside the application by the security management system and the ISSO. The benefit of this approach is that security services and capabilities are bundled together in a single Web services architecture rather than within stovepipe applications utilizing different standards, mechanisms, products, implementations, and configurations.

SECURITY MANAGEMENT AND PROVISIONING

With SOA, the challenge is to configure, maintain, and deploy consistent security policies across the Web services infrastructure. Web services are created and used many times over by many applications written and supported by many different programmers. Programs, other services, or human beings can execute these services from many places within the network. Security management and provisioning systems offload the security burden from developers and ensure consistent security application and management. Many systems calling Web services do not have the mapping capabilities to associate and authenticate requestors and repliers. Security Management Systems can provide this interface and mapping to META directories (AD, LDAP, native, etc.).

Complexity has traditionally been the enemy of security. A centralized security model utilizing common security policies and toolsets reduces the complexity and moves the security responsibility into the hands of the security professional. Centralized identity management and provisioning also provides for a single repository for authorized objects to the enterprise. It enables changes to be dynamically applied across the Web services enterprise for quick termination of accounts or dynamic change to move objects from one group policy to another.

LIBERTY ALLIANCE PROJECT AND PASSPORT

Today’s administrative and business environment calls for information sharing on an unprecedented scale, from government to business to citizen. Sharing and interoperating among agencies, businesses, and governments around the world create opportunities to simplify processes and unify work, as well as improve the overall performance of government. Secure interoperability, based on identity management solutions, enables substantial cost savings, streamlined processes, and faster communication of vital information to the benefit of governments and citizens of all nations. At the core of this revolution is the concept of identity management and the need for a standard that is open, interoperable, and decentralized. In addition, it must allow for privacy safeguards across all sectors.

The Liberty Alliance Project was established to address this need, and to tackle the twin issues of standards and trust. The Liberty Alliance is ushering in federated identity implementations that allow the public sector to find substantial benefits, including:

• Improved alliances, both within governments and between governments, through interoperability with autonomy
• Faster response time for critical communications
• Cost avoidance, cost reduction, and increased operational efficiencies
• Stronger security and risk management
• Interoperability and decreased development time

.NET PASSPORT

Passport is a suite of services for authenticating (signing in) users across a number of applications. The suite includes the Passport single sign-in service and the Kids Passport service.

.NET Passport Single Sign-In Service

The Passport single sign-in service solves the authentication problem for users by allowing them to create a single set of credentials that will enable them to sign in to any site that supports a Passport service (referred to as “participating sites”).

Passport simplifies sign-in and registration, lowering barriers to access for the millions of users with Passport accounts today. The objective of the Passport single sign-in service is to help increase customer satisfaction by allowing easy access without the frustration of repetitive registrations and forgotten passwords. As a part of the single sign-in service, if a user chooses to, he can store commonly used information in a Passport profile and, at his option, transmit it to the participating sites he visits. This reduces the barriers to acquiring customers because new users are not required to retype all of their information when they register at a new site. It also enables the sites they visit to customize...
and enhance their experience without having to prompt them for user information.

WEB SERVICES THREAT MODELS

Gartner predicts that by 2005, Web services will have reopened 70% of the attack paths against Internet-connected systems that were closed by network firewalls in the 1990s. Web services applications bypass traditional perimeter defenses and firewalls, and communicate through them over Hypertext Transport Protocol (HTTP) port 80 or Simple Mail Transport Protocol (SMTP). The threat then enters the protected internal network through the firewall and enters the application/Web services environment. The same attack scenarios that we have been seeing apply here as well:

- Traditional identity attacks, “Web services enabled”:
  - Identity spoofing
  - Eavesdropping
  - Man-in-the-middle attack
- Content-borne attacks:
  - SQL injection, LDAP injection, Xpath injection
- Operational attacks:
  - XML denial-of-service
  - Malicious or inadvertent attack

EVOLUTION OF FIREWALLS

Traditional network firewalls protect the physical boundaries of a network (category 1). The functionality provided by network firewalls is starting to expand to move up the OSI stack toward the application layer (category 2). There is a distinction between application level firewalls (category 3) and XML firewalls (category 4), and some situations may require some or all of these solutions.

Network Firewalls: Category 1

A network-level firewall sits at the doorstep of a private network as a guard and typically provides the following services:

- Monitors all incoming traffic
- Checks the identity of information requestors trying to access specific company resources
- Authenticates users based on their identities, which can be the network addresses of the service requesters or the security tokens
- Checks security and business policies to filter access requests and verify whether the service requestor has the right to access the intended resource
- Provides for encrypted messages so that confidential business information can be sent across the untrusted Internet privately

Application Firewalls: Category 2

Application-level firewalls will be required to provide edge shielding of servers running Web services exposed applications. They will focus on a small number of protocols—mainly HTTP and SMTP in the Web services world—and require a high degree of application awareness to filter out malicious XML constructs and encapsulations.

Such firewalls will be embedded in servers or act in conjunction with traditional firewalls, in much the same way that gateway-side content inspection is implemented today. Software-based solutions will not be successful on general-purpose Internet servers, but will be embedded in appliances or at the network level.

Application firewalls work in an interesting way: by learning what well-formed traffic to and from an application looks like and identifying the unexpected. To do this, Web application firewalls must inspect packets at a deeper level than ordinary firewalls. As with intrusion detection systems (IDSs), this is not a plug-and-play service; one must calibrate application firewalls carefully to reduce false positives without letting sneaky attacks through.

XML Firewalls: Category 3

XML firewalls can be used to protect corporations against the unique dangers and intrusions posed by Web services. These firewalls can examine SOAP headers and XML tags, and based on what they find, distinguish legitimate from unauthorized content. This entry now takes a look at how XML firewalls work, which vendors make them, and whether they are right for your organization today.

Traditional firewalls protect a network’s perimeter by blocking incoming Internet traffic using several different means. Some block all TCP ports except for port 80 (HTTP traffic), port 443 (HTTPS traffic), and port 25 (e-mail traffic). Some ban traffic from specific IP addresses, or ban traffic based on the traffic’s usage characteristics.

The problem with these firewalls when it comes to Web services is that, as a general rule, many Web services are designed to come in over port 80. So even if the service is a malicious one, the firewall will let it through. That is because traditional firewalls cannot filter out traffic based on the traffic’s underlying content—they can only filter on the packet level, not the content level. That is where XML firewalls come in. They are designed to examine the XML content of the
incoming traffic, understand the content, and based on that understanding, take an action—for example, letting the traffic in or blocking it.

XML firewalls typically work by examining SOAP message headers. The header may have detailed information put there specifically for the firewall to examine; and if so, the firewall can take an action based on that information. Even if the header does not have this information, XML firewalls can still take actions based on what is in the header. The header, for example, might have information about the recipients of the message, about the security of the overall message, or about the intermediaries through which the message has passed.

In addition, XML firewalls can look into the body of the message itself and examine it down to the tag level. It can tell if a message is an authorized one or is coming from an authorized recipient. If a federated ID system is involved, it can examine the SAML security token, and see if it trusts the token’s creator, and then take action based on that—for example, blocking traffic, sending it to a secure environment where it can be further examined, or allowing it to pass through.

XML firewalls have other methods of protection as well. They can understand metadata about the Web service’s service requestor as well as metadata about the Web service operation itself. They can gather information about the service requestor, such as understanding what role the requestor plays in the current Web service request. XML firewalls can also provide authentication, decryption, and real-time monitoring and reporting.

WEB SERVICES AND TRUST MODELS

The Web services trust framework ensures integrity in the authentication process, trusting who is vouching for whom. Good-faith trust is what contracts are about, and trust enters into a multitude of contractual arrangements. Through the Web services trust framework, the ebXML (electronic business XML) collaboration protocol profile and the agreement system enable one to make that kind of contractual arrangement machine-readable. One is agreeing to certain aspects of the interaction that one is going to have on a technical level, on a machine-machine level. Trust is built by explicitly specifying what it is one is going to do.

CONTRACTS AND LEGAL ISSUES

What are the compelling legal issues driving security within Web services? Be sure to consult with a legal professional throughout the life cycle of Web services development projects. In legal matters relating to Web services, being technically astute without being legally savvy could be trouble if the legal implication of a technical vulnerability is unknown—that is, in today’s environment where end-to-end security may not be technically feasible or not deployed (see Table 2). What security is required to contract online? Take a minimalistic view.

A contract can be defined as a promise or a set of promises the law will enforce. A contract does not depend on any signature; it depends on the will of the contracting parties. Also, some feel that a digital signature in itself is not analogous to an ink signature. Some claim that it is more difficult to forge ink on a paper signature repeatedly than steal an unsecured private key on a PC (but there is ongoing debate regarding this).

This is a can of worms and obviously left to the legal experts. It is important to note that the technical experts must confer with understanding regarding the risk, the value of the transaction or application, and the legal implications of binding contracts and holistic security. Enterprises must ensure and be able to demonstrate due diligence when conducting business on the Internet utilizing Web services.

CONCLUSION

While Web services attempt to simplify application security architectures and bundles with integrated standards, there are still many pieces that must be consciously designed and applied to equal a secure whole. Web services offers a lot of promise to developers of Web-based e-business applications or even the enhancement of traditional interfaces to legacy or even distributed systems. There is a bigger benefit to using this technology than not using it. However, security is still an issue and a challenge, and one needs to be aware of the potential security problems that might occur.

Holes, fillers, new standards, and solutions create a beacon with a clear and ever-resounding message: Proceed with caution!