APPLICATION PLATFORM STRATEGIES

Methodologies and Best Practices

burton GROUP™

Developing a Web Services Security Strategy

v1, September 30, 2005

AUTHOR:

Anne Thomas Manes
(amanes@burtongroup.com)

TECHNOLOGY THREAD:

Standards and Protocols

BURTON GROUP 7050 Union Park Center · Suite 510 · Midvale, Utah · 84047

P 801.566.2880
F 801.566.3611

www.burtongroup.com

Publishing Information

Burton Group is a research and consulting firm specializing in network and applications infrastructure technologies. Burton works to catalyze change and progress in the network computing industry through interaction with leading vendors and users. Publication headquarters, marketing, and sales offices are located at:

Burton Group
7050 Union Park Center, Suite 510
Midvale, Utah USA 84047-4169
Phone: +1.801.566.2880
Fax: +1.801.566.3611
Toll free in the USA: 800.824.9924
Internet: info@burtongroup.com; www.burtongroup.com

Burton Group's *Application Platform Strategies* service provides objective analysis of networking technology, market trends, vendor strategies, and related products. The information in Burton's *Application Platform Strategies* service is gathered from reliable sources and is prepared by experienced analysts, but it cannot be considered infallible. The opinions expressed are based on judgments made at the time, and are subject to change. Burton offers no warranty, either expressed or implied, on the information in Burton's *Application Platform Strategies* service, and accepts no responsibility for errors resulting from its use.

If you do not have a license to Burton's *Application Platform Strategies* service and are interested in receiving information about becoming a subscriber, please contact Burton.

Table of Contents

Introduction

Web services are hot. Although the technology that enables web services is only five years old, it has been almost universally adopted and implemented by the software industry.

For the purposes of this discussion, a web service is an application that exposes some type of business or infrastructure functionality through a language-neutral and platform-independent callable interface. In particular, the service exposes its functionality using the web services framework (WSF): It defines its interface using Web Services Description Language (WSDL), and it communicates using Simple Object Access Protocol (SOAP) and Extensible Markup Language (XML) messages. For more information on WSDL, SOAP, XML, and the WSF, see the following *Application Platform Strategies* documents:

- "The Advent of the Network Platform: Web Services Move into the IT Fabric" (overview)
- "Extensible Markup Language (XML) 2005: Core XML Standards and Their Impact on Business Development" (report)
- "Web Services Framework (WSF) Standards: The Interoperability Framework for Service-Oriented Architecture" (overview)

The WSF is hot because it enables easy interoperability between heterogeneous systems. The WSF is the latest incarnation of integration middleware, and unlike all previous middleware incarnations, it is natively supported by all major application platforms and a growing number of commercial off-the-shelf application systems.

Another reason why the WSF is hot is because it provides a technical foundation for service-oriented architecture (SOA). SOA is a style of application system design in which business functionality is implemented as shared, reusable services. In theory, after application functionality has been implemented as services, an organization will be able to mix and match these services and rapidly create new applications to support changing business requirements. Nearly every company Burton Group works with these days is at least talking about SOA, and most companies are developing plans to adopt SOA. See the following *Application Platform Strategies* documents for more information on SOA:

- "Service-Oriented Architecture: Developing the Enterprise Roadmap" (overview)
- "VantagePoint 2005-2006 SOA Reality Check" (overview)
- "Building the Business Case for Service-Oriented Architecture Investment" (Methodologies and Best Practices [MBP] document)

To summarize, web services enable open, flexible, and adaptable systems. But with openness comes increased risk. Without proper security protections, a web service can expose vulnerabilities that may cause dire consequences for an enterprise. When deploying web services, an enterprise must implement an appropriate set of security safeguards to counter numerous threats. These threats are summarized in the "Problem Statement" section of this MBP document.

Although security precautions can be implemented on a service-by-service basis, a much more cost-effective approach to securing web services is to develop an enterprise-wide web services security strategy. Security is a tricky subject, and an enterprise should not assume that the average developer deeply understands the nature of all threats that jeopardize an application system. Therefore, security should not be left to the whim of the developer. Whenever possible, security should be managed, configured, and coordinated by security professionals.

A Systematic, Comprehensive Security Strategy

Web services security should be part of an organization's information security strategy. As described in the *Security and Risk Management Strategies* overview, "A Systematic, Comprehensive Approach to Information Security," information security should be based on a sound understanding of the organization's business with a goal to reduce business risks. (Risks are a function of threats that may exploit vulnerabilities and cause consequences.)

Security is not just about technology. An effective security strategy requires both strong technologies and comprehensive governance processes. Security precautions should be commensurate with business risks, and precautions should be implemented consistently throughout the environment. For more information about security governance strategies, see the *Security and Risk Management Strategies* overview, "Security Governance for the Enterprise."

Externalizing Security Functionality

One way to reduce costs and to improve the consistent implementation of security precautions throughout the environment is to externalize generic (yet extremely complex) security functionality from application code as much as possible through the deployment of a shared security infrastructure. Generic security functionality includes services such as authentication, authorization, auditing, and cryptographic processing. Using an application security framework, developers can configure their applications to make use of shared security services according to specified policies. Security frameworks are described in detail in the *Application Platform Strategies* overview, "Application Security Frameworks: Protecting Applications Consistently."

Most web services infrastructure products include built-in security frameworks that enable security administrators to configure service protections interactively, via an administrative console, or declaratively, using configuration files. In many cases, developers don't need to write even one line of code to implement these generic security functions.

According to an enterprise architect at a Fortune 50 financial conglomerate, externalization of security functionality can generate the following benefits:

- **Reduced costs:** On average, generic security functionality, including authentication, authorization, and auditing, consumes approximately 15% of business application

development and ongoing information technology (IT) operations costs. By removing these nonbusiness functional requirements from application code, a large organization could cut hundreds of millions of dollars each year from its IT budgets.

- **Faster time to market:** Externalization of security functionality reduces application complexity and therefore reduces the time it takes to develop and test applications.
- **More consistent and reliable auditing:** Defining and implementing one auditing process at the global level rather than thousands of auditing processes (one for each application) will greatly reduce complexity and risks. Individual applications would only need to perform simple auditing to assert compliance with the enterprise infrastructure.
- **Reduced risk:** By centralizing security responsibility to people with more expertise, security precautions can be applied more consistently and on a level commensurate with business risk.

Risk Tradeoffs of Externalized Security

Of course, there are tradeoffs associated with externalizing and centralizing security functionality. For one thing, such processes generate a concentration of risk. If an organization develops one hundred authentication modules, it's unlikely that all of them will be bulletproof; however, it is also unlikely that all modules will be compromised—an attacker would need to research and develop one hundred different attacks to break them all. On the other hand, if an organization develops just one authentication module and reuses it in one hundred applications, all applications could be compromised by a single attack. Care must be taken to ensure that the security infrastructure (and interfaces to the security infrastructure) go through appropriate levels of security testing and change control for the risk involved. For more analysis of this problem, see the *Security and Risk Management Strategies* overview, "Risk Aggregation: The Unintended Consequence."

Another challenge presented by externalizing security is that the centralized security group may not have the necessary resources it needs to adequately assess the security risks associated with a complex distributed application system. Development teams can't completely abdicate responsibility for an application's security. Senior application architects should be trained in the relevant security disciplines so that they are, at least to some degree, security professionals. Likewise, all developers should be trained in secure coding and testing techniques in order to reduce code vulnerabilities.

Application security starts with secure code, and secure code requires that application development teams think about security from the beginning of the software development lifecycle (SDLC). Ultimately, it is the application that has the most knowledge of its own environment and of the valid and invalid states that can occur. Often, only the application can perform custom authorization based on the context of an individual transaction, or in some cases, the aggregation of multiple transactions. And only the application developer can ensure that his or her own program code is free of vulnerabilities. For more information about secure coding techniques, see the *Security and Risk Management Strategies* report, "Application Security: Everybody's Problem."

Policy-Based, Layered Defenses

The *Security and Risk Management Strategies* Reference Architecture Template, "Information Security Technology Model," describes a policy-based, layered security model that supports centralized or coordinated management of security policies. The model, shown in Figure 1, describes three interdependent control systems that manage and coordinate the security infrastructure. Policy management authorities (PMAs) enable centralized definition and management of security policies. They include management consoles and provisioning systems. Policy enforcement points (PEPs) ensure that policies are properly enforced at runtime. Policy decision points (PDPs) determine how security policies apply to specific situations.

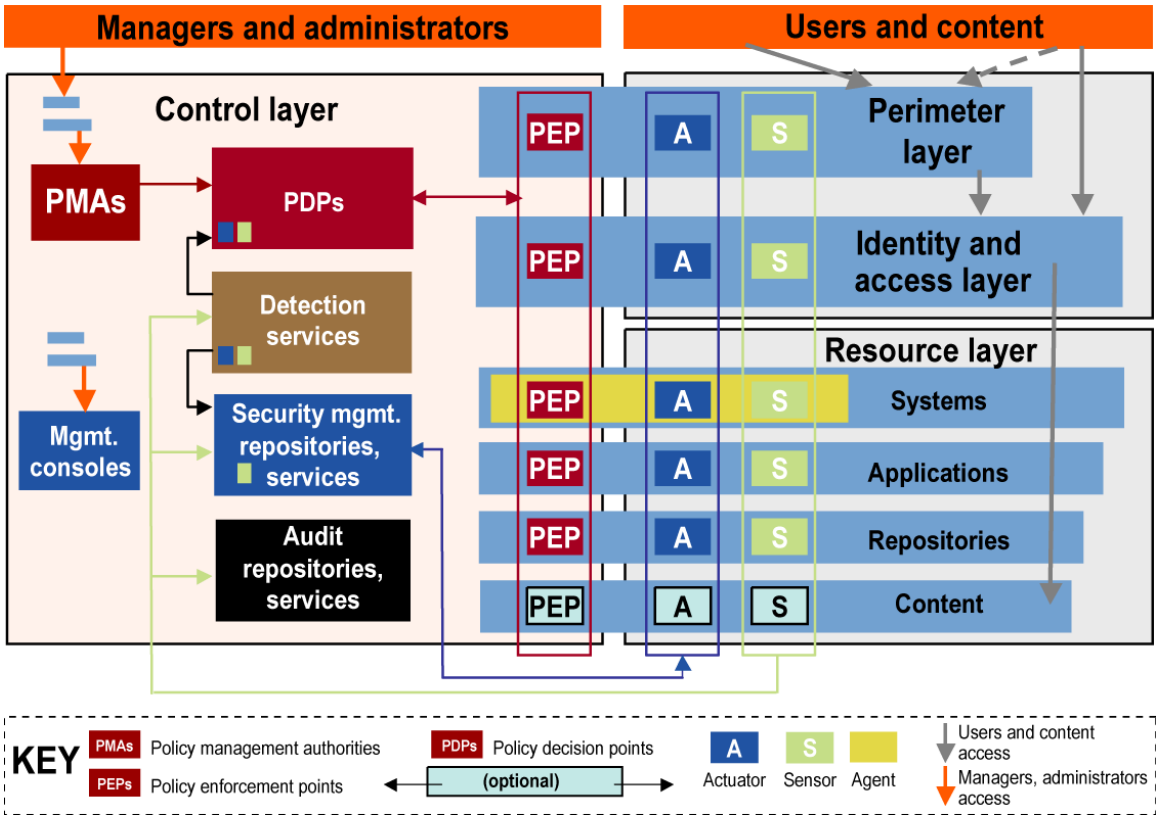

Figure 1: *Information Security Technology Model*

PEPs are distributed throughout the environment. They combine perimeter defense tactics with identity-based access control. Perimeter controls are typically deployed at centralized network access control points (in the perimeter layer) and often operate invisibly to applications. Identity-based controls are deployed at centralized and distributed access points (in the identity and access layer) and within protected resources (in the resource layer). For guidance on where and when to deploy security controls, see the *Security and Risk Management Strategies* and *Identity and Privacy Strategies* Reference

Architecture Technical Positions. For example, the *Security and Risk Management Strategies* Reference Architecture Technical Position, "System Placement and Trust Mechanisms," covers perimeter protection decisions based on whether a host is mobile and what network zone of trust it occupies.

Problem Statement

Because nearly all application platforms now support WSDL and SOAP, the WSF enables easy interoperability. But it also exposes application systems to a number of security threats. The Web Services Interoperability Organization (WS-I) has produced an analysis of the security threats associated with web services in a requirements document entitled "Security Challenges, Threats and Countermeasures Version 1.0." Burton Group recommends that all web services security architects read this WS-I document. Table 1 summarizes the various security threats that jeopardize web services environments, as identified by WS-I.

ID	Name	Description
T-01	Message alteration	The message information is altered by inserting, removing or otherwise modifying information created by the originator of the information and mistaken by the receiver as being the originator's intention. There is not necessarily a one-to-one correspondence between message information and the message bits due to canonicalization and related transformation mechanisms.
T-02	Confidentiality	Information within the message is viewable by unintended and unauthorized participants (e.g., a credit card number is obtained).
T-03	Falsified messages	Fake messages are constructed and sent to a receiver who believes them to have come from a party other than the sender. For example, Alice sends a message to Bob. Mal copies some (or all of) it and uses that in a message sent to Bob who believes this new action was initiated by Alice. This overlaps with T-01. The principle is that there is generally little value to saying a message has not been modified since it was sent unless we know who sent it.
T-04	Man in the middle	A party poses as the other participant to the real sender and receiver in order to fool both participants (e.g., the attacker is able to downgrade the level of cryptography used to secure the message). The term "man in the middle" is applied to a wide variety of attacks that have little in common except for their topology. Potential designs have to be closely examined on a case-by-case basis for susceptibility to anything a third party might do.

T-05	Principal spoofing	A message is sent which appears to be from another principal (e.g., Alice sends a message which appears as though it is from Bob). This is a variation on T-03.
T-06	Forged claims	A message is sent in which the security claims are forged in an effort to gain access to otherwise unauthorized information (e.g., a security token is used which wasn't really issued by the specified authority). The methods of attack and prevention here are essentially the same as T-01.
T-07	Replay of message parts	A message is sent which includes portions of another message in an effort to gain access to otherwise unauthorized information or to cause the receiver to take some action (e.g., a security token from another message is added). Note that this is a variation on T-01. Like "man in the middle," this technique can be applied in a wide variety of situations. All designs must be carefully inspected from the perspective of what could an attacker do by replaying messages or parts of messages.
T-08	Replay	A whole message is re-sent by an attacker.
T-09	Denial of service (DoS)	Amplifier Attack: attacker does a small amount of work and forces system under attack to do a large amount of work. This is an important issue in design and perhaps merits profiling in some cases.

Table 1: *Security Threats that Jeopardize Web Services (Source: WS-I "Security Challenges, Threats and Countermeasures Version 1.0")*

Additional threats that are not defined in the WS-I document include:

- Content-borne threats, such as viruses and worms
- Schema poisoning, which can cause the message payload to be altered during validation or can cause DoS attacks
- Injections of illegitimate scripts or data in the message payload that may compromise security processes (e.g., Structured Query Language [SQL] injections)
- Misuse of services for fraudulent purposes

What's Different About Web Services?

Anyone who is familiar with information security will quickly realize that these threats aren't unique to web services. In fact, nearly all of these threats (with the possible exception of schema poisoning) apply equally to traditional web applications. So what's different about web services? Why is it that an organization needs to develop a different security strategy for web services than it uses for web applications?

In fact, a web services security strategy should build on an organization's web application security strategy—especially because many web applications are now using web services as back-end resources. See the *Identity and Privacy Strategies* Reference Architecture Template, "Web Access Management Services," for guidance on designing a web application security strategy.

But web services applications are slightly different from web and portal applications. For one thing, access to a web service may not be constrained to a single point of entry. Most organizations manage and control access to web applications using a portal system, which supplies a comprehensive profile-based security framework that supports access controls based on rules, roles, and content. These portal systems could be used to implement security controls for web services, but only for interactive clients. And that would address only a small percentage of web services traffic. The WSF is designed to support application-to-application integration. No humans need to be involved.

So just how does one implement a comparable security control system for non-interactive web services?

Today, many organizations use the WSF only for simple point-to-point application integration, and for these applications, simple security measures (such as transport-level authentication and message protection) are quite adequate. But as organizations begin to exploit the power of web services to support SOA and composite application development, the increased complexity of the environment will require more sophisticated security strategies.

A composite web services application may comprise numerous services written using various programming languages, deployed on multiple application platforms, and running on numerous operating systems and hardware systems. The application invokes these services in sequence, according to a predefined execution script or based on dynamic business rules. Message traffic may travel directly between services, or it may be routed through multiple intermediaries, which perform functions such as data transformation, content-based routing, and business activity monitoring. Traffic may originate from within corporate boundaries, or it may travel to and from external organizations.

To complicate matters further, the services within a composite application often are encapsulated legacy applications. In many cases, these applications have integrated resource-level security controls that use different means to represent security information, such as user identity and privileges. Composite applications must manage the mapping of user credential information from service to service and still maintain references back to the original requesting entity for auditing purposes. This requirement to maintain multiple credentials with a message (originator, intermediaries, and final destination) adds a level of complexity to the process rarely seen in web applications.

And finally, web services applications often use loosely coupled connections between services. Loose coupling can increase the vulnerability of these applications to threats such as falsified messages and man-in-the-middle and replay attacks.

Overview of Web Services Security Requirements

The security requirements for a given web service interaction are determined by the risk assessment associated with the specific web service. From a general perspective, though, the web services infrastructure must support the following security requirements:

- **Entity identification and authentication:** A web services interaction involves at least two participating entities (a sender and a receiver), and it may involve additional mediation entities (intermediaries). The web services infrastructure must support mechanisms that allow entities to identify and authenticate each other. Identification is the process through which an entity proves its identity by presenting a set of claims. Authentication is the process through which the identity claims are verified.

- **Authorization:** After the participating entities have been identified and authenticated, it may be necessary to determine whether the parties are entitled to exchange the specific information in question. Authorization is the process through which an entity gains permission to use a resource, such as a document, service, or specific combination of data.

- **Data origin identification and authentication:** The receiving entity may require information about the source of all or part of a message. The information may have been created by the initial sender or an intermediary, or it may have been created by another entity not directly participating in the web services interaction. Data origin identification is the process through which an entity provides proof regarding the origin of a piece of data, and data origin authentication corroborates the origin claim.

- **Data integrity in motion:** The web services infrastructure must be able to verify that message data has not been changed or damaged in transit, either through accidental or nefarious means.

- **Data integrity at rest:** Message data may be persisted in one or more places during the course of a web services interaction. The web services infrastructure must ensure full-fidelity integrity of data as it is stored and retrieved or serialized and de-serialized.

- **Data confidentiality in motion:** The web services infrastructure must be able to ensure that unauthorized entities cannot view or access sensitive information (all or part of a message) while it is in transit.

- **Data confidentiality at rest:** The web services infrastructure must be able to ensure that unauthorized entities cannot view or access sensitive information (all or part of a message) while it is in memory or persisted.

- **Message uniqueness:** The web services infrastructure must be able to detect duplicate messages, which may be used in replay and DoS attacks.

- **Message validation and content scanning:** The web services infrastructure must be able to validate messages and message attachments to ensure that they contain appropriate data and do not contain attack vectors, such as viruses and worms, or compromising message content, such as SQL injections.

- **Auditing:** A business often requires an audit trail of application interactions for purposes such as regulatory compliance, service-level compliance, and billing. It may be necessary to trace an individual message (and its adherence to security policy) years after the fact. It is particularly important to audit changes made to security policies.

- **Monitoring:** The web services infrastructure should support monitoring of web services interactions to enable detection of potential security breaches and fraud.

- **Management and administration:** Security administrators need tools for defining security policies, configuring PEPs, provisioning users, and managing identities, roles, permissions, and keys. A web services security infrastructure should integrate with existing public key infrastructure (PKI) and identity management (IdM) systems.

- **Trust management:** Web services interactions often require communications that cross security domains. The web services infrastructure must support the ability to establish and recognize trust relationships and to map credentials from one trust domain to another.

- **Federated administration:** Even though the best practice is to have one enterprise infrastructure supporting security, each significant business domain must be able to configure its own security policies and participate in a workflow process to negotiate security policies with its internal and external partners.

Web Services Security Governance

Perhaps most important of all, an organization must address web services security governance. Governance refers to the processes that an enterprise establishes to ensure that things are done right, where "right" means in accordance with best practices, architectural principles, federal regulations, and other determining factors. Web services security governance refers to the processes used to ensure that web services implement the appropriate amount of security functionality that is commensurate with risk and corporate policy.

Web services security governance involves three steps:

1. Define security policies.
2. Deploy a web services security infrastructure that enables adherence to these policies.
3. Institute a set of formal processes and procedures that verify compliance with these policies.

Step 1: Define Security Policies

Security policies codify the rules and guidelines that dictate what security precautions must be taken for each application or service. Policies relate to issues such as:

- How do you assess the risk associated with a particular service?
- What security precautions must be implemented to mitigate those risks?
- What's the maximum overhead that the security precautions can impose?
- What tools and technologies should be used to implement the security precautions?
- Who is responsible for implementing the security precautions?
- Who is responsible for ensuring that the security precautions have been implemented properly?
- What documentation must be generated for auditing compliance with security policies?

Step 2: Deploy a Web Services Security Infrastructure

A web services security strategy should build on existing corporate security infrastructure systems, such as network security, PKI, IdM, and access control systems. It should not require a completely new infrastructure. But web services security requires additional technologies specific to web services traffic. Web services security infrastructure product options are discussed in the "Solution Options" section of this MBP document. Solutions that support externalization and centralized management of security functionality typically make it easier to implement web services security governance.

Step 3: Institute Compliance Processes

Governance is about ensuring that things are done right; therefore, an enterprise must institute processes that verify that the systems comply with corporate policies. Policies aren't of much use if no one follows them.

Compliance processes may be manual or automatic. Examples of compliance processes include:

- Design reviews
- Code reviews
- Code profiling
- Compatibility testing
- Compliance testing
- Staging
- Approval
- Runtime monitoring

For more information about security governance strategies, see the *Security and Risk Management Strategies* overview, "Security Governance for the Enterprise."

Topology Options

A comprehensive web services security strategy requires security policy enforcement at numerous points throughout the environment. In order to support a comprehensive layered defense system, PEPs should be deployed in the perimeter layer and the identity and access layer.

Perimeter Layer Security

A web services security strategy should build on existing perimeter security systems, such as firewalls, proxy servers, virtual private networks (VPNs), intrusion detection and response systems (IDRSs), Internet Protocol (IP) address filtering, and content filtering systems. These traditional network-based perimeter security systems provide the first layer of defense against attacks and intrusions. Other than message validation, traditional perimeter systems don't address the security requirements outlined in the previous section of this MBP document, but they do limit the potential traffic entering the web services infrastructure, and therefore they simplify the challenges associated with securing the environment.

For information about traditional perimeter security best practices, see the following documents from the *Security and Risk Management Strategies* Reference Architecture:

- "Perimeter Security Model" (Template)
- "Perimeters and Zones" (Technical Position)

Identity and Access Layer Security

Identity and access layer PEPs enforce security policies at a point of access to a resource. These PEPs implement access controls based on the requesting entity's identity or based on other considerations such as transaction state or application context. An identity and access layer PEP can support most web services security requirements, including authentication, authorization, encryption, signature processing, provisioning, credential mapping, message scanning, message validation, denial-of-service detection, auditing, and monitoring.

Identity and access layer PEPs are typically deployed throughout an environment, providing additional layers of defense using the following patterns:

- Centralized PEPs
- Intermediary PEPs
- Endpoint PEPs

Figure 2 illustrates a layered defense strategy using identity and access layer PEPs.

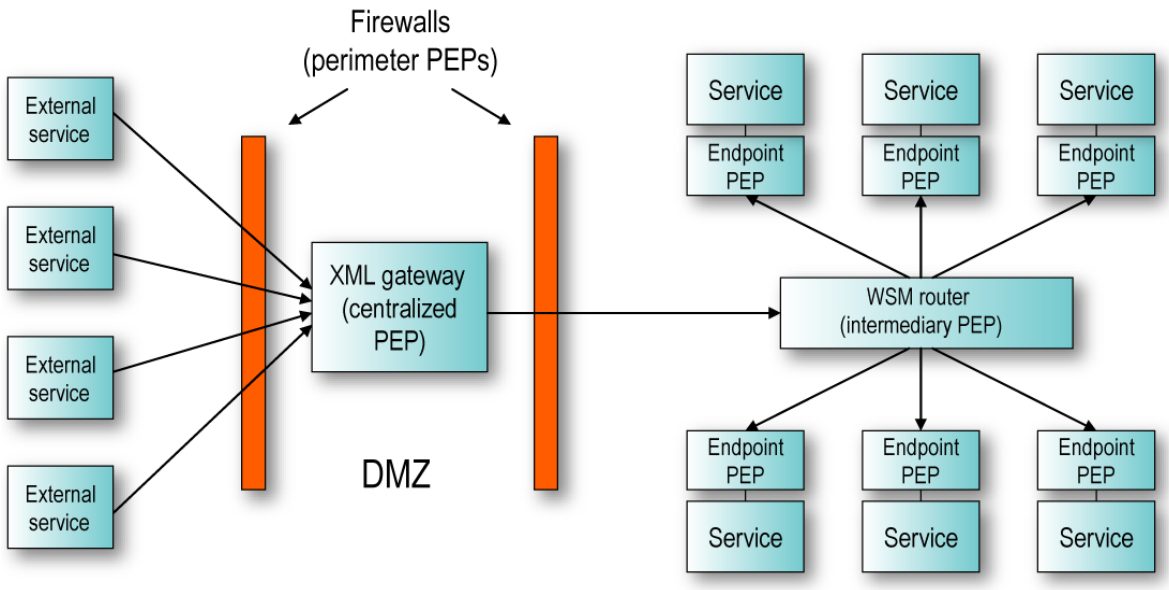

Figure 2: *A Layered Defense Strategy Using Identity and Access Layer PEPs*

Centralized PEPs

A centralized PEP implements identity-based access control at a single point of entry to a system. For example, a portal security framework provides a centralized PEP for web applications. An XML security gateway, which is typically deployed in a demilitarized zone (DMZ), provides similar control for web services traffic. The "XML Security Gateways" section of this MBP document provides a brief overview of XML security gateways.

Intermediary PEPs

Intermediaries in a web services infrastructure support a variety of management and mediation services, such as routing, message transformation, protocol switching, exception management, service-level monitoring, and business activity monitoring. In addition to management and mediation, intermediaries can also perform a variety of security functions, including authentication, credential mapping, authorization, encryption, signature processing, content filtering, and auditing.

Products that supply intermediaries include XML security gateways, web services management (WSM) systems, and enterprise service bus (ESB) systems. The "Web Services Management" section of this MBP document describes the security capabilities of WSM products, and the "Web Services Platforms" section describes the security capabilities of ESB products. For more information on web services intermediaries, see the following *Application Platform Strategies* reports:

- "Web Services Management: Gaining Control of Distributed Services"
- "Enterprise Service Bus: EAI in Transition"

Endpoint PEPs

The last line of defense in a security infrastructure is a PEP deployed at the service endpoint. A service endpoint is implemented using a web services platform (WSP). Some WSPs, such as Apache Axis and Systinet Server, are available as stand-alone frameworks, although more often, a WSP is bundled with other products (such as a superplatform, an application server, or an ESB). There are dozens of WSPs available, supporting a variety of application programming languages.

A WSP includes a security framework that, at a minimum, supports authentication functionality. Typically, a WSP uses a pluggable authentication system. For example, most Java platforms use the Java Authentication and Authorization Service (JAAS), and Microsoft .NET relies on the pluggable authentication system supported by Internet Information Services (IIS). These pluggable authentication systems allow the WSP to support multiple authentication mechanisms and to relegate authentication and authorization decisions to an external PDP, such as a Lightweight Directory Access Protocol (LDAP)-compliant directory or an IdM system.

A WSP also supports extensions using an interception model, which is described in the *Application Platform Strategies* overview, "Application Security Frameworks: Protecting Applications Consistently." Using this extensible interception model, developers can embed a PEP within the service endpoint to address additional web services security requirements, such as authorization, auditing, encryption, signature processing, and content validation.

Most WSPs provide a built-in framework for implementing endpoint PEPs. The "Web Services Platforms" section of this MBP document describes the built-in security capabilities of WSP products. Developers can also use third-party frameworks that plug into the WSP interception model. Most WSM solutions provide cross-platform plug-in frameworks, and there are a number of product-independent plug-in libraries available. The "WS-Security Libraries" section of this MBP document describes the product-independent plug-in libraries.

For more information on WSPs, see the following *Application Platform Strategies* documents:

- "Selecting a Java Web Services Platform: An Evaluation Framework" (overview)
- "The Java Superplatforms: Comparing IBM, Oracle, and BEA" (report)
- "SAP NetWeaver: Potential Wildcard in the Superplatforms Arms Race" (report)
- "The Microsoft Superplatform: Setting the Bar in the Superplatform Arms Race" (report)
- "Enterprise Service Bus: EAI in Transition" (report)

Message Security Options

Message security can be implemented at one of two levels:

- Transport-level security
- Application-level security

Transport-Level Security

Participating entities in a web services interaction can use the integrated security features of a network transport protocol to address many web services security requirements, including entity identification and authentication, data integrity and confidentiality in motion, and message uniqueness.

Web services infrastructure products typically support interoperable security using the following transport-level security mechanisms:

- **HTTP Authentication (Request for Comments [RFC] 2617):** Hypertext Transfer Protocol (HTTP) supports one-way entity identification and authentication using a simple username-based challenge/response protocol.
- **SSL/TLS (RFC 2246):** The Secure Sockets Layer (SSL) and Transport Layer Security (TLS) protocols support mutual entity identification and authentication using X.509 certificates and cryptography. SSL/TLS also supports data integrity, data confidentiality, and message uniqueness.

Transport-level security mechanisms are relatively easy to use, but they provide only coarse-grained data integrity and confidentiality protections. For example, SSL/TLS supports signing and encryption of a complete message, but it does not support signing and encryption of selected portions of a message. In addition, transport-level security mechanisms provide protection only during transit between two communicating ports, and these mechanisms don't necessarily support end-to-end protection between two communicating applications. For example, the original sender's transport-level authentication information is not propagated across an intermediary. Also, message data is decrypted and therefore vulnerable to confidentiality and integrity breaches while at rest in the intermediary. Consequently, transport-level mechanisms alone may not be sufficient for applications that communicate via intermediaries.

For information on transport-level security features, see the following documents:

- *Identity and Privacy Strategies* Reference Architecture Technical Position, "User Authentication"
- *Network and Telecom Strategies* report, "The Changing Face of SSL-based Remote Access"
- *Identity and Privacy Strategies* overview, "Public Key Infrastructure: Architecture and Concepts"
- *Identity and Privacy Strategies* Reference Architecture Technical Position, "Public Key Infrastructure"

Application-Level Security

Application-level security mechanisms, which manage the security context at the application layer, provide protection between two logical application endpoints, regardless of the number of intermediaries in the path. Figure 3 illustrates the difference between transport-level security, which provides point-to-point protection, and application-level security, which provides end-to-end security.

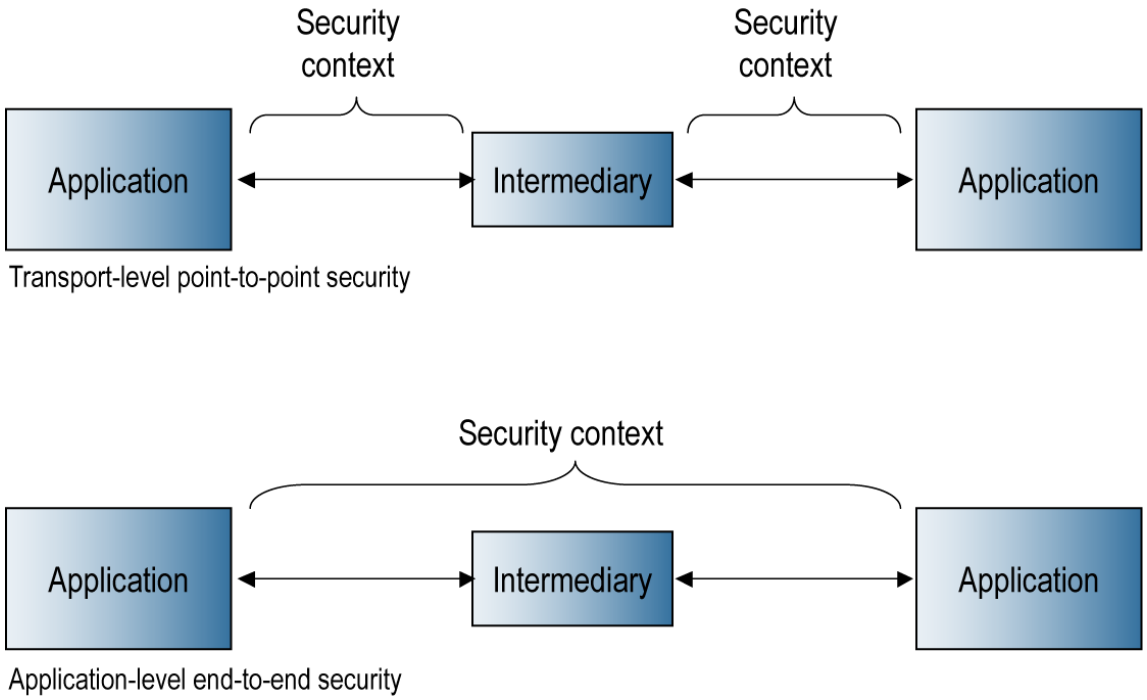

Figure 3: *Point-to-Point vs. End-to-End Security*

WS-Security

The Web Services Security: SOAP Message Security 1.0 (WS-Security 2004) specification defines a standard, interoperable mechanism for implementing application-level security in web services. WS-Security supports entity identification and authentication, data origin identification and authentication, data integrity and confidentiality, and message uniqueness. WS-Security also provides a foundation for authorization and auditing.

WS-Security defines extensions to the SOAP messaging protocol that enable SOAP messages to carry security tokens in a SOAP header. A security token is a mechanism for encapsulating security information that may be used for authentication, authorization, signing, encryption, session management, and other functions. WS-Security supports username tokens, binary tokens (such as X.509 certificates and Kerberos tickets), and XML tokens (such as Security Assertion Markup Language [SAML] assertions and Rights Expression Language [REL] licenses). WS-Security 2004

also defines a processing model for signing and encrypting message content using XML Signature and XML Encryption.

WS-Security supports an extensible set of security tokens, mechanisms, and algorithms, affording tremendous flexibility and control. A participating entity may supply multiple tokens representing different types of claims and assertions. For example, one token can provide proof of identity, another token can provide proof of data origin, and third token can provide an attribute assertion indicating spending limits. Each intermediary can add tokens to the message, enabling externalization of security functionality as well as maintenance of an audit trail.

WS-Security Standards

WS-Security 2004 1.0 was ratified by the Organization for the Advancement of Structured Information Standards (OASIS) in April 2004, and since then, the specification has been broadly adopted by web services infrastructure vendors. See the "Solutions Options" section of this MBP document for information about products that support WS-Security.

In addition to the core WS-Security specification, OASIS has ratified four token profile standards:

- Web Services Security (WSS) Username Token Profile 1.0: Supports entity identification and authentication using a username and password
- Web Services Security X.509 Certificate Token Profile: Describes how to use the X.509 authentication framework with WS-Security
- Web Services Security SAML Token Profile: Describes how to exchange SAML 1.1 assertions with WS-Security
- Web Services Security REL Token Profile: Describes how to exchange REL licenses with WS-Security

The OASIS technical committee is also developing a token profile for Kerberos, but this specification has not yet been ratified.

WS-*

WS-Security provides the core framework that enables basic message security. It also provides the foundation for the following extended security specifications, collectively known as WS-*:

- WS-Policy: A metadata language that provides the means to define policies (such as security policies)
- WS-SecurityPolicy: A policy assertion language for specifying security policies in WS-Policy
- WS-PolicyAttachment: Mechanisms for associating policies with a web service
- WS-MetadataExchange: A protocol that allows communicating parties to exchange metadata, including WSDL, XML Schema, and WS-Policy descriptions

- WS-Trust: A specification for a security token service (STS), which is a web service that can distribute and validate security tokens. WS-Trust provides a foundation for cross-domain federation
- WS-SecureConversation: Extends WS-Trust and provides the means to establish and maintain an extended security session between two or more communicating parties
- WS-Federation: Extends WS-Trust and defines a set of profiles for enabling federation across trust realms

For detailed information about WS-Security and the WS-* extensions, see the *Security and Risk Management Strategies* overview, "WS-*: A Composable Architecture for Web Services Security." Also see the *Application Platform Strategies* overview, "VantagePoint 2005-2006 SOA Reality Check," for information about the WS-Policy framework.

Solution Options

More than 30 vendors provide web services security solutions. These products fall into seven categories:

- WS-Security libraries
- Web services platforms
- Web services management
- XML security gateways
- XML VPNs
- Web services fraud detection
- Web services single sign-on (SSO) and federation

Table 2 provides a summary matrix of WS-Security-related product functionality. The first column indicates support for the OASIS WS-Security 2004 standard. The subsequent columns represent support for XML Encryption, XML Signature, Username token, X.509 token, SAML token, REL token, Kerberos token, and the WS-Policy framework, respectively. Note that support for the Kerberos token profile is based on a preliminary working draft specification, which is subject to change. A "Y" indicates support. An "N" indicates no support. An asterisk ("*") indicates partial support. A "Q" indicates the next calendar quarter in which the vendor expects to deliver support. (For example, "Q4" indicates that the vendor plans to deliver the feature in the fourth quarter of 2005, and "Q1" indicates that the vendor plans to deliver the feature in the first quarter of 2006.) A "06" indicates that the vendor plans to deliver this feature sometime in 2006.

	WSS 2004	Encryption	Signature	Username	X.509	SAML	REL	Kerberos	WS-Policy
WS-Security libraries									
Apache WSS4J 1.0	Y	Y	Y	Y	Y	Y	N	N	N
RSA BSAFE Secure-WS 1.0	Y	Y	Y	Y	Y	N	N	N	N
Sun XML and Web Services Security (XWS-Security) 2.0	Y	Y	Y	Y	Y	Y	N	N	N
VeriSign Trust Services Integration Kit (TSIK)	Y	Y	Y	Y	Y	N	N	N	N
Web services platforms									
BEA Systems WebLogic 9.0	Y	Y	Y	Y	Y	Y	N	N	Y
Cape Clear ESB 6.1	Y	Y	Y	Y	Y	N	N	N	N
Fiorano ESB 4.0	Y	Y	Y	Y	Y	Y	N	N	N
IBM WebSphere 6.0	Y	Y	Y	Y	Y	*	N	*	N
IONA Artix 3.0.2	*	*	N	Y	N	N	N	Y	N
Microsoft Web Services Enhancements (WSE) 2.0 SP3	Y	Y	Y	Y	Y	N	N	Y	Y
Oracle 10g R3	Q4	Q4	Q4	Q4	Q4	Q4	N	N	N
SAP NetWeaver 6.40	Y	Y	Y	Y	Y	Q4	N	N	06
Sonic ESB 6.1	Q1	Q1	Q1	Q1	Q1	N	N	N	Q1
Systinet Server 6.0	Y	Y	Y	Y	Y	N	N	N	Y
TIBCO BusinessWorks 5.2	Q1	Q1	Q1	Q1	Q1	N	N	N	N
webMethods Glue 6.0	Y	Y	Y	Y	Y	Y	N	N	N
Web services management									
Actional SOAPstation 5.5	Y	Y	Y	Y	Y	Y	N	Q4	Y
AmberPoint 4.3	Y	Y	Y	Y	Y	Q4	N	N	Q4
Blue Titan 3.5	Y	N	Y	Y	Y	Y	N	N	Y
Hewlett-Packard SOA Manager 2.0	Y	Y	Y	Y	Y	N	N	N	N
Infravio X-broker 4.7.1	Y	Y	Y	Y	Y	N	N	N	N
Oracle WSM 4.03	Y	Y	Y	Y	Y	Y	N	N	N
SOA SW Service Mgr 3.0	Y	Y	Y	Y	Y	Y	N	N	Y
XML security gateways									
Cisco AON	Y	Y	Y	Y	Y	Y	N	Y	N
DataPower XS40	Y	Y	Y	Y	Y	Y	Y	Y	Y
Forum Sentry 5.1	Y	Y	Y	Y	Y	Y	N	Y	N
Intel Guardian (Sarvega)	Y	Y	Y	Y	Y	Y	N	N	N
Layer 7 SecureSpan Gateway 3.1	Y	Y	Y	Y	Y	Y	N	N	Y
Reactivity Secure XML	Y	Y	Y	Y	Y	Y	N	Y	N
Vordel VordelSecure	Y	Y	Y	Y	Y	Y	N	*	N

XML VPNs									
Layer 7 SecureSpan Bridge 3.1	Y	Y	Y	Y	Y	Y	N	N	Y
SOA SW XML VPN	Y	Y	Y	Y	Y	Y	N	N	Y
Web services authentication and entitlements									
Computer Associates *e*Trust TransactionMinder	Y	Y	Y	Y	Y	Y	N	N	N
Entrust Identification and Entitlements Services (IES)	Y	N	N	Y	Y	Y	N	N	N
IBM Tivoli FIM	Y	*	Y	Y	Y	Y	N	N	N

Table 2: *WS-Security Product Functionality Matrix as of August 2005*

WS-Security Libraries

A WS-Security library is an implementation of a WS-Security provider that may be plugged into a WSP.

A WSP provides the tools and runtime frameworks that developers use to build and deploy web services—the endpoints in the web services infrastructure. All SOAP-compliant WSPs support an extensible interceptor framework that enables the WSP to implement application-level security functionality within the endpoint. When the WSP receives a message containing a SOAP header element, such as a WS-Security header, it calls an interceptor "handler" to process the header. This interceptor handler uses a WS-Security provider to perform functions such as authentication, XML encryption and decryption, and signature generation and verification.

A WS-Security library provides a callable application programming interface (API), and it supports invocation via an interceptor handler. Most Java WSPs support the standard interceptor framework defined in the Java API for XML-based Remote Procedure Call (JAX-RPC) specification. Many WSPs also support custom interceptor frameworks that enable finer control and automation.

WS-Security libraries include:

- Apache WSS4J
- RSA Security BSAFE Secure-WS
- Sun XWS-Security
- VeriSign TSIK (which is now an Apache Software Foundation incubator project)

Web Services Platforms

At a minimum, a WSP's security framework supports transport-level authentication using HTTP and SSL/TLS and message protection using SSL/TLS. Most WSPs also supply integrated support for WS-Security 2004, although they support different features with varying degrees of automation. Nearly all WSPs support XML encryption and signatures, and the Username and X.509 token profiles. Very few WSPs support the SAML, REL, and Kerberos token profiles or the SwA profile.

Unfortunately, the WS-Security 2004 specification does not define a standard process for configuring security policies; therefore each product uses a different process.

WSPs include:

- BEA WebLogic Server (which also provides the foundation for BEA's integration product family, AquaLogic)
- Cape Clear ESB
- Fiorano ESB
- IBM WebSphere Application Server (WAS)
- IONA Artix
- Microsoft .NET and WSE
- Oracle Application Server
- SAP NetWeaver Application Server
- Sonic ESB
- Systinet Server
- TIBCO Software BusinessWorks
- webMethods Fabric

Web Services Management

A WSM solution provides a platform-independent management and control environment for a heterogeneous web services infrastructure. A WSM solution typically includes an administrative console and a set of agents that can be deployed throughout the infrastructure. These agents monitor message traffic and collect information used for service level agreement (SLA)-compliance tracking, error detection and resolution, business activity monitoring, and security monitoring. These agents can also perform a variety of infrastructure functions, such as security enforcement, reliable message delivery, content-based message routing, message transformation, and other types of mediation.

WSM agents, which operate as SOAP intermediaries, may be implemented as stand-alone software proxies, application server filters, or SOAP interceptor handlers. Therefore, WSM solutions can provide centralized, intermediary, and endpoint PEPs.

WSM solutions typically support more security options than the built-in security frameworks in a WSP. For example, many WSM solutions support credential mapping, provisioning, content filtering, and security violation monitoring. WSM solutions also frequently support security of non-SOAP traffic. They can offload expensive XML processing, such as encryption, signatures, and transformations, from the application servers hosting web services. WSM solutions often work in concert with XML security gateways for added acceleration and key management services. One advantage of using a WSM solution is that it provides a single management and administration tool and consistent features and functions for the entire web services environment. (See the *Application Platform Strategies* report, "Web Services Management: Gaining Control of Distributed Services," for more information.)

Most WSM solutions support WS-Security 2004, and all provide interactive consoles for configuring security policies across the enterprise. Nearly all WSM solutions support XML encryption and signature processing, as well as the Username, X.509, and SAML token profiles. Very few WSM solutions support the REL and Kerberos token profiles or the SwA profile.

WSM solutions include:

- Actional SOA Command and Control, which comprises a control console and policy server (Looking Glass), lightweight monitoring agents (Active Agents), and feature-rich brokering agents (SOAPstation)
- AmberPoint Management Foundation (AMF)
- Blue Titan Network Director, which comprises a console and policy server (Blue Titan Manager), management agents (Blue Titan Control Points), a registry (Blue Titan Registry), and a set of infrastructure services (Blue Titan Fabric Services)
- Hewlett-Packard SOA Manager
- Infravio Ensemble, which comprises a console and policy server (X-console), a WSM agent (X-broker), and a registry (X-registry)
- Oracle Web Services Manager (formerly known as Oblix COREsv)
- SOA Software Service Manager, which comprises a console and policy manager (Console), a set of management agents (Management Point), and a set of client interface tools (Gateway)

XML Security Gateways

An XML security gateway is a SOAP intermediary, similar to a WSM agent, that focuses specifically on managing and enforcing security. Gateways can be used to implement centralized and intermediary PEPs. One vendor (Vordel) also provides a configuration that supports endpoint PEPs (VordelDirector).

Gateways typically support more security features than WSM agents, including key management, authentication, authorization, encryption, signature processing, credential mapping, message scanning, message validation, attack detection, denial-of-service detection, auditing, monitoring, and other functions. Some gateways support only security functionality; others support additional intermediary functionality, such as reliable message delivery, content-based message routing, and message transformation. An XML gateway is typically packaged as a hardware appliance, although some vendors supply the technology in other hardware form factors or as software. The hardware configurations feature XML and cryptographic acceleration features. Most hardware appliances have received Federal Information Processing Standard (FIPS) 140-2 certification.

All XML security gateways support WS-Security 2004, and all provide interactive consoles for configuring security policies across the enterprise. All gateways support XML encryption and signatures, and the Username, X.509, and SAML token profiles, and a few support the Kerberos and SwA profiles. Most gateways also support security of non-SOAP XML traffic.

XML security gateways include:

- Cisco Application-Oriented Networking (various hardware modules)
- DataPower XS40 XML Security Gateway (hardware appliance)
- Forum Sentry (hardware appliance, blade, Peripheral Component Interconnect [PCI] card, or software)
- Intel XML Guardian Gateway (formerly Sarvega—hardware appliance)
- Layer 7 SecureSpan Gateway (hardware appliance, blade, or software)
- Reactivity Secure XML Infrastructure (three hardware appliances)
- VordelDirector (software) and VordelSecure (hardware appliance or software)

XML VPNs

An XML VPN provides client-side provisioning services, such as credential management, PKI provisioning, identity bridging and federation, signature and encryption processing, web services SSO, and transaction auditing. A client application uses the XML VPN to dynamically obtain permission to access a service. The XML VPN can also assist the client in obtaining the appropriate credentials it needs to access the service.

XML VPNs support WS-Security 2004, and all provide interactive consoles for configuring provisioning policies. All XML VPNs support XML encryption and signature processing, as well as Username, X.509, and SAML token profiles.

Two vendors provide XML VPNs—Layer 7 and SOA Software. An XML VPN works in conjunction with an XML security gateway. The Layer 7 SecureSpan Bridge works with the Layer 7 SecureSpan Gateway, which also operates as a stand-alone gateway. SOA Software XML VPN comprises a provisioning server (Controller) and an XML security gateway (Appliance). SOA Software does not sell its gateway appliance as stand-alone product.

Web Services Fraud Detection

WSM solutions, XML security gateways, and XML VPNs all support web services monitoring for SLA management and policy compliance tracking, but none of these products provides any specific capabilities for detecting fraud. One vendor, Service Integrity, provides a web services business intelligence system, called SIFT, which provides a unique capability to detect fraud.

Using an architecture similar to that of a WSM solution, SIFT includes an administrative console and set of agents that can be deployed throughout the infrastructure. But unlike WSM agents, SIFT agents do not act as PEPs—they simply monitor message traffic and passively collect information. SIFT also includes a customizable business intelligence dashboard that provides real-time visibility into web services systems for performance optimization, policy compliance tracking, auditing, metering and billing, business activity monitoring, fraud detection, and other uses.

Web Services SSO and Federation

Many companies use a web access management (WAM) system, such as Computer Associates *e*Trust SiteMinder, Entrust GetAccess, IBM Tivoli Access Manager, or Oracle COREid, to enable SSO for web traffic, but these WAM systems don't necessarily extend the same capabilities to web services. Most WSPs, WSM, XML gateways, and XML VPNs can use these WAM systems for single-message authentication and authorization, but they can't use them for SSO or for cross-domain federation without some additional technology.

Three of these vendors—Computer Associates, Entrust, and IBM—have developed extensions to their WAM systems to enable web services SSO and federation. The Entrust and IBM solutions enable open integration using WS-Trust. The Computer Associates solution supplies a plug-in agent. All three products support WS-Security 2004 1.0 with Username, X.509, and SAML tokens. Oracle plans to add support for web services federation via WS-Trust to COREid Federation (formerly known as Oblix SHAREid) in Q4 2006.

Web services SSO and federation products include:

- Computer Associates *e*Trust TransactionMinder 6.0
- Entrust IES
- IBM Tivoli Federated Identity Manager

Case Studies

As part of the research for this MBP document, Burton Group interviewed a number of companies regarding their development of a web services security strategy. One of the most interesting points of information learned from the interview process is how few people currently use WS-Security. An informal poll taken in the application security cross-cutting concerns track at Catalyst North America in July 2005 found that less than 1% of those surveyed use WS-Security. Most companies currently rely on transport-level security to secure web services traffic. Burton Group interprets this finding to mean that most companies are still using web services only for point-to-point communications, and service-oriented architecture (SOA) is still a future endeavor. These companies have not yet started to use intermediaries or to build composite applications.

State Law Enforcement Company

One state law enforcement company has implemented a security strategy based on a restricted, encrypted network, a centralized PEP, and transport-level security mechanisms. This company has a web services application, implemented using Apache Axis and running on Apache Tomcat, that provides access to state and federal law enforcement information.

This service is made available to local law enforcement agencies. Client applications communicate with this service using SOAP over Hypertext Transfer Protocol Secure (HTTPS) and SSL mutual

authentication. Client requests also embed user-ID information in the application payload for audit and recall purposes.

All requests enter the system through an Intel XML Guardian Gateway appliance. The Intel device scans the message and performs schema validation, content checking, and threat detection. Next, the device authenticates the user using SSL mutual authentication, and then it calls out to Active Directory using Directory Services Markup Language (DSML) to make a complex authorization decision. The authorization process considers who is making the request, what information is being requested, and whether the requestor has the proper certifications to obtain the requested information. Based on the results of the authorization request, the Intel device then routes the request to the appropriate web service(s). A single request may be split into multiple requests to back-end services hosted locally or on various federal systems.

Long-Term Strategy

For the moment, this company is content with the system as it is, but the security architects recognize that at some point they will need to adopt a message-level security strategy to enable more flexibility via mediated communications. (Currently, all mediation occurs in the Intel appliance.) The architects are concerned, though, that adoption of WS-Security and SAML will first require them to implement a comprehensive IdM solution.

Financial Services Company

One financial services company implements web services security using centralized and endpoint PEPs and a combination of transport-level and message-level security mechanisms. All external communications are routed through SOA Software's XML VPN. The company hosts the XML VPN Appliance within its DMZ. The XML VPN Controller is hosted by a third-party service provider.

All external partners must be invited to join the network before they may access the company's financial services. Upon acceptance of the invitation, the XML VPN Controller automatically provisions the external partner and propagates the user's rights and privileges to the XML VPN Appliance. The partner may then immediately start using the services.

All external requests are routed through the XML VPN Appliance, which authenticates the request using SSL mutual authentication and verifies that the requestor has rights to access the service. The XML VPN Appliance then adds a WS-Security header to the request message. It maps the user information to a WSS Username token, adds a timestamp, and signs the message. At that point, the XML VPN Appliance forwards the message to the requested service.

The XML VPN Appliance and the service authenticate using SSL mutual authentication. In fact, all internal SOAP communications require SSL mutual authentication. The Username token is used for auditing purposes. In some cases, the requested service may also use the Username token for application-specific authorization.

Support for WS-Security is somewhat sporadic in the back-end systems, which makes management and coordination of the environment somewhat challenging. The back-end systems are predominantly implemented using IBM WAS 5.1 and Microsoft .NET. Because IBM WAS 5.1 doesn't support WS-Security 2004, the company developed its own framework to implement support for WS-Security. Most .NET applications use Microsoft WSE 2.0, although some also use the in-house framework. A few older services don't use WS-Security 2004; instead, they use proprietary headers based on the SOAP Security Extensions: Digital Signature (SOAP-DSIG) specification, which predates WS-Security and was never ratified.

Long-Term Strategy

This company plans to continue to use an XML security gateway for centralized policy enforcement. It plans to upgrade the capabilities of this PEP, though, such that the gateway becomes a trusted SAML server that supports cross-domain federation. In the future, the gateway will attach SAML tokens to incoming messages in place of the current Username tokens.

The company also plans to adopt a WSM product to replace its sporadic use of WS-Security frameworks and to enable consistent management of security policy throughout the environment. The company has not yet begun evaluating WSM vendors. Point-to-point service communication will continue to use SSL mutual authentication. Applications will use the SAML tokens for auditing and application-specific authorization decisions.

In addition, the company plans to implement a middleware infrastructure that will enable the company to expose new and legacy application systems as abstract services that can be accessed using SOAP over HTTP or SOAP over WebSphere MQ. The company has evaluated a number of ESB products for the task, and at the moment, it seems to be leaning toward IBM WebSphere Message Broker. Regardless of the ESB selected, the company plans to implement endpoint security using a WSM product.

Telecommunications Company

One telecommunications company implements web services security using intermediary policy enforcement points (PEPs) and a combination of transport-level and message-level security mechanisms based on Vordel's VordelDirector. The company selected Vordel's solution because of its tight integration with Entrust GetAccess, which the company uses for its browser-based business-to-business (B2B) applications.

Figure 4 shows a diagram of the company's web services security architecture. All inbound and outbound B2B traffic flows through a DMZ "bridge" comprising VordelDirector, Entrust GetAccess Server, and an in-house developed broker. The VordelDirector gateway system consists of a set of distributed agents, which act as intermediary PEPs, and replicated VordelDirector services, which act as policy decision points (PDPs). VordelDirector supports automatic load balancing and failover, thereby avoiding a single point of failure.

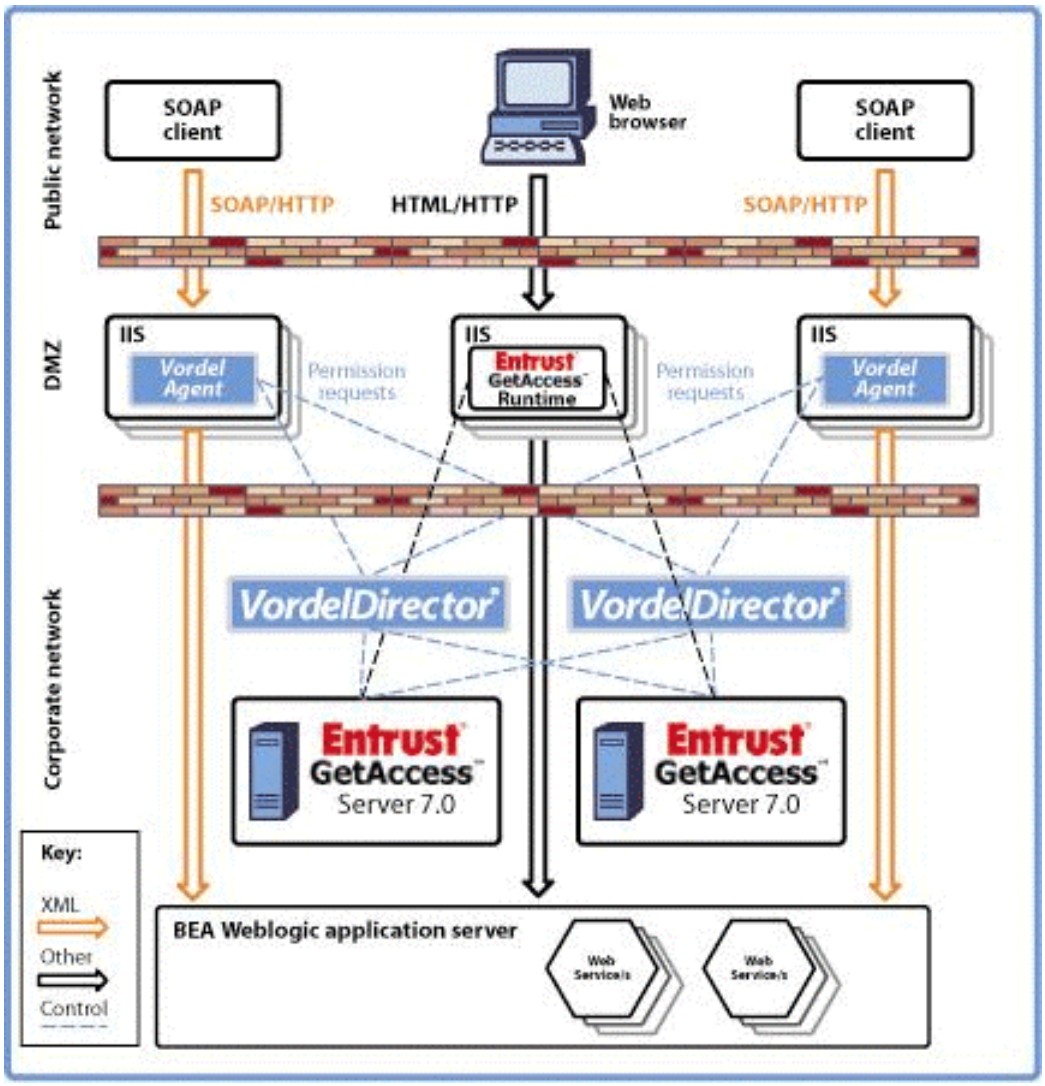

Figure 4: *A Telecommunication Company's Web Services Security Architecture (Source: Vordel)*

Each B2B SOAP message is initially routed to a Vordel agent, which is deployed as a plug-in in a Microsoft IIS web server. All traffic uses SSL to encrypt the channel and WS-Security for dual-factor authentication. Users must supply a Username and an X.509 signature. The Vordel agent performs message validation and virus checking (using McAfee Olympus). Then it sends a policy decision request to VordelDirector, which uses Entrust IES (using SAML and WS-Trust) to query Entrust GetAccess Server for an authentication and authorization decision. If GetAccess approves access, then the message is forwarded to the broker for processing.

The back-end environment is heterogeneous—comprising some SOAP services (.NET and WebLogic) and some non-SOAP services (Vitria and mainframe). Each type of service uses a different user management system and speaks with different protocols. When the broker gets a request, it maps it into something that can be understood by the back-end system. The broker maps the inbound user credentials to HTTP Authentication or mainframe System Authorization Facility (SAF) credentials, and it transforms the message into the appropriate format and protocol.

Long-Term Strategy

This company plans to maintain its current DMZ and external access policies, although it eventually plans to add support for SAML-based federation. Federation plans are currently on hold because Entrust GetAccess does not yet support interoperability with other SAML providers.

The company also plans to adopt WS-Security with either Username or SAML tokens for all internal service communications, although this plan will also require encapsulation of the non-SOAP services with web services framework (WSF) interfaces and a coordinated IdM strategy.

Thomson Prometric

Thomson Prometric, a division of the Thomson Corporation, is a provider of testing and assessment solutions for government, academic, and professional organizations. Thomson Prometric is an early adopter of orchestration and composite application systems. For example, to support the student scheduling process, Thomson Prometric has used an orchestration engine, Microsoft BizTalk Server, that invokes multiple back-end applications in response to a single request. Each request passes through one or more mediators during processing.

In order to deal with the complexity of interacting with multiple legacy systems while maintaining the security of the system and the privacy of the student, Thomson Prometric implemented a security strategy based on centralized, intermediary, and endpoint PEPs with message-level security. The centralized PEPs are implemented using a cluster of Reactivity XML Security Gateway appliances. The intermediary and endpoint PEPs are implemented using Actional SOAPstation agents.

Figure 5 shows an overview of Thomson Prometric's edge infrastructure architecture. All external requests enter the system through the Reactivity gateway. Each request contains two sets of credentials encoded in a WS-Security header. The first set of credentials represents the application submitting the request. The second set of credentials represents the user.

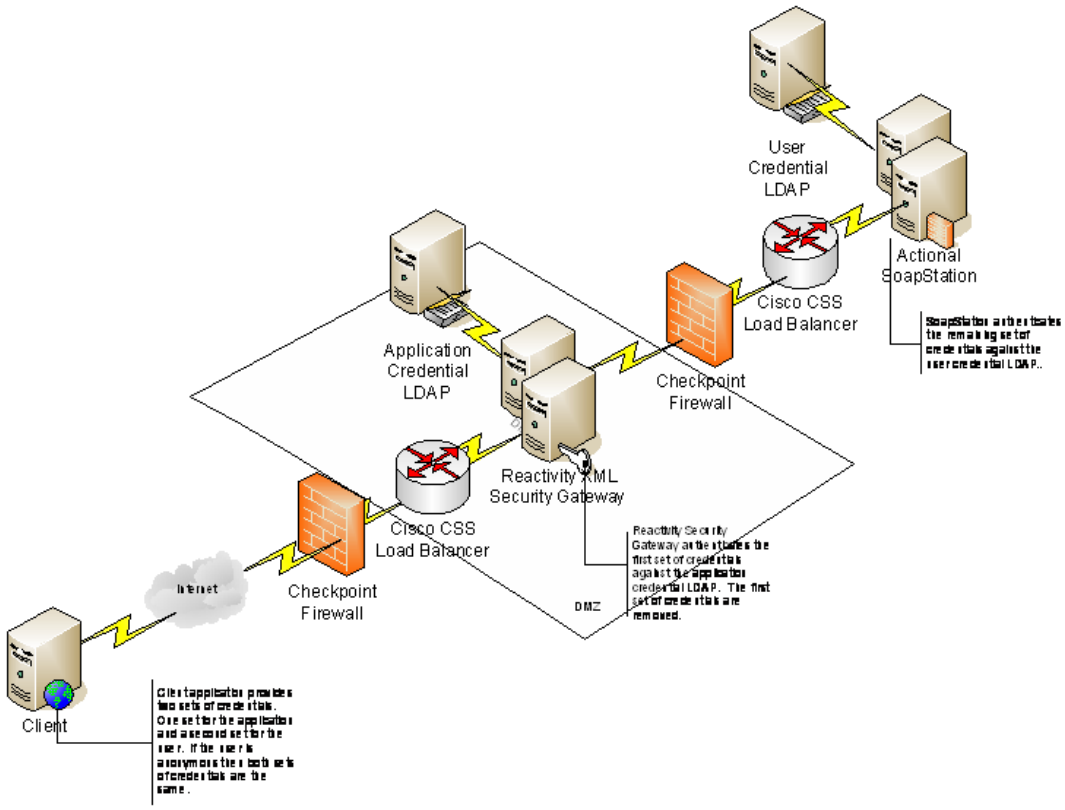

Figure 5: *Thomson Prometric's Edge Infrastructure Architecture (Source: Thomson Prometric)*

The Reactivity gateway performs a number of functions for each incoming message:

- It authenticates the application, and if the application has been properly provisioned, the gateway strips off the application credentials.
- It validates the message integrity using digital signatures, verifies the message's compliance with its XML Schema description, and inspects the message for rogue content, such as SQL injection.
- It transforms the message payload and the user credentials for seamless interoperability between Prometric's testing brokerage and its 4,000 worldwide testing centers.

The Reactivity gateway then routes the message to the appropriate web service endpoint via an intermediary Actional SOAPstation PEP, which authenticates the user credentials, provides monitoring and fault diagnosis capabilities, and then forwards the message to the web service. The last mile between the SOAPstation agent and the web service is protected by ensuring that the web service accepts messages only from the SOAPstation IP address range using IP filtering.

Because Thomson Prometric uses orchestration, the first web service endpoint is only the beginning of the application process. For example, if a user submits a request to schedule an exam, the message will be initially routed to the scheduling service, which in turn forwards the request to the BizTalk orchestration engine. BizTalk Server then invokes a number of other services in a choreographed sequence to accomplish the scheduling process. Each service endpoint invoked by BizTalk Server is protected by a SOAPstation agent that provides monitoring and fault diagnostic capabilities. The final response message is returned to the Reactivity gateway, which transforms and protects the message before routing it back across the Internet to the calling application.

In the future, Thomson Prometric plans to deploy additional Reactivity gateways within the infrastructure to act as intermediary PEPs and to take over responsibility for the user authentication process. Actional SOAPstation agents will still be used as endpoint PEPs to provide monitoring and fault diagnostics.

For more information about this case study, see the slides presented by Christopher Crowhurst at Burton Group's Catalyst North America conference in July 2005.

Fortune 50 Financial Conglomerate

One large financial conglomerate has designed a secure communications infrastructure that protects all web services traffic, both internal and external. The security infrastructure relies on centralized and intermediary PEPs, with transport-level and application-level security mechanisms.

All point-to-point interactions use SSL to maintain a secure channel, and they use SSL for mutual authentication. (Each service endpoint has its own certificate.) In addition, each message carries a SAML assertion that is used for auditing.

All traffic must pass through a security tier to get on and off the communications infrastructure. The security tier comprises a Forum Sentry gateway appliance and AmberPoint Management Foundation (AMF) agents. A message enters the channel through the Forum gateway, which performs SSL mutual authentication and machine-based authorization. It then generates a SAML authentication assertion and inserts it into a WS-Security header. The Forum appliance then routes the request to the appropriate service endpoint via an AMF agent, which monitors all traffic for security breaches, errors, and service-level violations. When the service receives the request, it logs the SAML token for the audit trail.

The communications path to and from services via the security tier is ensured using SSL mutual authentication. Services may not communicate directly; they may only communicate with the Forum appliance or an AMF agent.

Long-Term Strategy

This company is contemplating the creation of a comprehensive SOA-based security infrastructure because it anticipates that it can save hundreds of millions of dollars per year by externalizing and

centralizing the authentication, authorization, and auditing processes. The goal of this effort would be to allow the SOA infrastructure to eventually become the "notary" for the system. The communications infrastructure, by itself, will ensure the security of the environment rather than leaving security to the whim of individual developers or business units.

The company will continue to use SSL for authentication, and it will also use SAML for auditing. The company anticipates using SAML for authorization and attribute assertions. It is planning to evaluate IBM, Computer Associates, Hewlett-Packard, and one or two other SAML-based authorization systems.

Thus far, the company has relied exclusively on SSL to support data integrity and confidentiality, although it now has business requirements that dictate adoption of message-level encryption and signatures. The company has completed proof-of-concept trials and plans to deploy message-level protection systems in its production processes in the near future.

Recommendations

A comprehensive web services security strategy should address all the threats and requirements identified in the "Problem Statement" section of this MBP document. The team responsible for defining an organization's security strategy should have a deep understanding of these threats and requirements, as well as the technologies that can address them. The team should also understand these threats from a business perspective and should design security protections commensurate with risk assessments. (For more information, see the *Security and Risk Management Strategies* report, "Building Secure Applications: How Secure Do You Want to Be Today?")

How Much Should You Spend on Security?

Security designers should always consider the costs associated with implementing security protections. For example, it probably doesn't make sense to invest $10 million in added security for an application when the consequence of a security breach will cost at most $1 million.

Another important cost consideration is the latency impact of compute-intensive activities, such as validation, encryption, and signature processing. These latency issues can be mitigated by using optimized XML processing solutions, such as hardware appliances. In any case, an organization should establish a policy regarding the maximum additional latency that security processes may impose on transactions, and the web services security strategy must operate within these constraints.

The cost of deploying a centralized security infrastructure should be viewed as an investment that will pay for itself over time. As indicated in the "Externalizing Security Functionality" section of this MBP document, externalizing security functionality reduces development costs and quickens time to market. Centralized security management also reduces administrative and operations costs and aids

corporate governance efforts through better visibility, more consistent auditing, and more consistent application of corporate policies.

What Topology Should You Use?

A web services security strategy requires a layered defense approach. The first line of defense should be based on traditional network-based perimeter security solutions, including firewalls, proxy servers, VPNs, and content-filtering systems.

The second line of defense should be a centralized identity and access layer PEP—most likely implemented using an XML security gateway. It's not a coincidence that all five companies in the case studies presented in this MBP document use an XML security gateway as the foundation for their web services security strategies. A centralized identity and access layer PEP can also be implemented using a WSM solution.

The last line of defense should be distributed identity and access layer PEPs at the service endpoints. An endpoint PEP can be deployed either within the service endpoint or as a proxy in front of it. If the PEP is deployed as a proxy, then separate measures should be taken to ensure that the PEP cannot be circumvented. Effective measures include using IP filtering or requiring SSL mutual authentication.

If web services interactions are mediated, then additional PEPs should be deployed at each intermediary. In other words, no web services interactions should be permitted that don't include a PEP.

What Should These PEPs Do?

The specific functions performed by a PEP should be determined by application requirements and by corporate security policies based on the risk assessment and commensurate surety levels associated with the particular web services interaction. Typically, though, a PEP will (at a minimum) authenticate the communicating parties and audit the interaction. All PEPs also should monitor interactions and feed information to management dashboards for SLA monitoring, intrusion detection, and fraud detection. Centralized PEPs typically also perform operations such as credential mapping, message validation, content scanning, and threat detection. The location of authorization processing is frequently dependent on the requirements of the application.

How Should You Implement These PEPs?

Centralized and intermediary PEPs should be implemented using XML security gateways and/or WSM agents. Although most WSPs provide integrated frameworks for implementing PEPs, an organization should consider standardizing on a WSM solution and using its agents to implement all endpoint PEPs. Each WSP supports a slightly different set of security features and capabilities, and each environment is managed and configured differently. In many cases, the WSP encourages

developers to be responsible for service security policy rather than externalizing it to security professionals. Using the WSPs' built-in PEP capabilities will rapidly result in administrative chaos.

A WSM solution provides a consistent, centralized environment for defining and administering web services security policy throughout the enterprise. Many WSM solutions are also integrated with some of the XML security gateway products for centralized management. A WSM solution also provides a more comprehensive security solution than the WSP built-in frameworks. For example, most WSM solutions support automatic auditing, monitoring, credential mapping, and content filtering, and they support message mediation and security of non-SOAP traffic.

When Do You Need to Use WS-Security?

As demonstrated by the case studies, transport-level security mechanisms are quite effective at supporting authentication and message protection requirements—particularly for point-to-point interactions. Because HTTP Authentication is rather weak, organizations should restrict its use to web services with only the very lowest surety requirements, and all messages containing usernames and/or passwords should be encrypted using SSL. SSL mutual authentication is a much stronger mechanism and adequate for most applications. Applications with high surety requirements may require multifactor authentication tests. These additional identification factors (e.g., keys, signatures, and biometric tokens) may be exchanged using WS-Security.

SSL mutual authentication can be adequate for mediated web services interactions, but only if PEPs are deployed within all endpoints and intermediaries. SSL authentication can ensure that all endpoint communications are routed through approved message paths, ensuring that PEPs cannot be circumvented.

Although many web services interactions may not require WS-Security for authentication, organizations should consider using WS-Security to maintain user information (a Username or SAML token) for auditing and authorization purposes. This tactic is especially useful for mediated web services interactions, which otherwise would lose track of the identity of the original requester when using transport-level authentication. As a general rule, transport-level authentication should be used for endpoint authentication rather than user authentication. If user information must be captured in audit trails, or if authorization decisions need to be made based on user identity, the user information should be carried in a WS-Security header.

Some applications may need to use XML encryption to support certain high-surety business requirements. SSL encryption does not support data confidentiality at rest—for example, while the message is being processed by a SOAP intermediary (including all identity and access layer PEPs). If the consequences of a security breach are severe enough, it may be appropriate to encrypt sensitive portions of the message using XML encryption. The best practice for application-level message protection is to encrypt only those elements that carry sufficient risk such that transport-level security is inappropriate. The rest of the message should be protected using SSL.

Use of XML signatures is likely to become more common in the near future, especially for mediated interactions. Data origin identification and authentication requires use of XML signatures, and many applications will require XML signatures for legal purposes (e.g., auditing and non-repudiation). The security infrastructure will also use XML signatures to combat threats, such as replay and man-in-the-middle attacks and falsified messages. Again, the best practice is to sign only those elements that require protection rather than the entire SOAP envelope or body.

What Else Do You Need?

At a minimum, a comprehensive web services security strategy requires a traditional perimeter security zone, an XML security gateway, and a WSM solution. But these systems represent only the surface. A web services security infrastructure must build on the corporate security infrastructure, and underneath it all, solid PKI and IdM solutions must be in place. Widespread use of SSL for mutual authentication demands seamless and pervasive key provisioning and management. Meanwhile, seamless integration across numerous legacy security domains requires at least automatic credential mapping, if not centralized management and administration of user identities.

As organizations progress from simple point-to-point integration to composite applications and complex orchestration scenarios, the need for centralized trust management and federation will grow. Organizations should start planning now to adopt SAML 2.0 and WS-Trust. Some IdM vendors, including IBM and Entrust, have already delivered preliminary federation solutions based on WS-Trust.

Another useful component of a comprehensive web services security strategy is a web services registry (WSR). A WSR provides a central, categorized index of information about all services in an SOA. It supplies a virtual system of record that brings order to an extremely distributed, diverse, heterogeneous environment. Service providers use a WSR to advertise their services to potential consumers. Service consumers use a WSR to discover services that match their requirements and to locate the metadata and policies associated with each service endpoint.

A WSR also provides a foundation for SOA governance, and with it, web services security governance. The registration process provides a single point of control for the institution of compliance and approval processes. An organization can use these approval processes to ensure that all services implement appropriate PEPs in adherence to corporate policies.

Many WSM solutions offer seamless integration with a WSR, extending the reach of SOA governance to cover the complete service lifecycle—from application design and registration to provisioning and runtime operations. A WSR can use a WSM to help manage the service provisioning process, and a WSM can use a WSR to maintain information about managed services and to learn about new services that need to be provisioned and managed. Together, they streamline runtime monitoring, service provisioning, change management, and version control.

Vendors that supply WSRs include Blue Titan, Infravio, SOA Software, and Systinet. For more information about WSRs, see the *Application Platform Strategies* report, "Web Services Registry: The Foundation for SOA Governance." ●

Conclusion

The web services framework (WSF) has taken a lot of heat over the past five years regarding a perceived lack of security. Many organizations have cited this issue as the major impediment to their adoption of web services. And yet, more than a year after WS-Security was ratified, very few organizations are using it. Instead, most organizations rely on a combination of Extensible Markup Language (XML) security gateways and transport-level mechanisms to secure web services traffic. These protections are quite adequate for simple point-to-point interactions, but as organizations progress to true service-oriented architecture (SOA), a more comprehensive security strategy will be required. ●