The Intersection of Web Services and Security Management:
A Service-Oriented Security Architecture

A META Group White Paper

For Web services to succeed in extending the remarkably successful document Web into a trusted business services Web that reliably spans the globe, its designers must apply service-oriented security architecture concepts to architect a new trust management Web that can be federated across administrative domains and spheres of trust. The goal is not to replace the diverse security infrastructures already deployed, but to enable them to interoperate end-to-end.
# The Intersection of Web Services and Security Management:
A Service-Oriented Security Architecture

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Executive Summary</td>
<td>1</td>
</tr>
<tr>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>The Invention of the Service-Oriented Architecture</td>
<td>3</td>
</tr>
<tr>
<td>Evolving Beyond the Limits of the Web to Web Services</td>
<td>4</td>
</tr>
<tr>
<td>Concept of a Service-Oriented Security Architecture</td>
<td>5</td>
</tr>
<tr>
<td>Definition of a Web Services Security Model</td>
<td>7</td>
</tr>
<tr>
<td>Trusted Communication</td>
<td>8</td>
</tr>
<tr>
<td>Trusted Service</td>
<td>9</td>
</tr>
<tr>
<td>Trust Management Web</td>
<td>9</td>
</tr>
<tr>
<td>The First Step: WS-Trust</td>
<td>9</td>
</tr>
<tr>
<td>The Road Ahead: Filling in the WSSM</td>
<td>12</td>
</tr>
<tr>
<td>Comparison to Traditional Security Architectures</td>
<td>13</td>
</tr>
<tr>
<td>Impact on Security Vendors and Markets</td>
<td>14</td>
</tr>
<tr>
<td>Bottom Line</td>
<td>15</td>
</tr>
</tbody>
</table>
The Intersection of Web Services and Security Management: A Service-Oriented Security Architecture

Integration & Development Strategies
Nick Gall and Earl Perkins

Executive Summary
As businesses evolve toward increased interdependence via the Internet and the Web, they face a steadily rising need for better coordination across organizational and business boundaries. Spanning the increasingly diverse technologies that must interoperate and the growing number of administrative and security domains that must cooperate to enable such coordination requires an interoperable, federated approach to security.

To evolve existing corporate and regional communications networks into a global communications meta-network, the original designers of the Internet created the concept of a service-oriented architecture (SOA). An SOA federates existing technologies and policies by virtualizing their behavior into a unified service model.

For the emerging Web services architecture to succeed in extending the existing (and remarkably successful) document Web into a trusted business services Web that reliably spans the globe, its designers must apply SOA concepts to architect a new trust management Web that can be federated across administrative domains and spheres of trust.

This White Paper will discuss the SOA shaping this evolving trust management Web and the emerging Web services security standards that define it. It will also discuss how these standards complement existing security policies, processes, standards, and technologies.

Introduction
For several years, leveraging the Internet and the Web has been a key goal of many CIOs. The Internet is unquestionably changing the way we do business — indeed the way we live our lives. The value of the Internet increases not only with every additional business or consumer that plugs into it (the so-called network effect), but also with every new technology that layers over it (e.g., Web, e-mail, instant messaging, peer-to-peer file sharing, voice over IP) and every new technology that lies beneath it (e.g., optical and wireless networks). The Internet is becoming the universal way to span geographical, national, administrative, and even business and interpersonal boundaries, to globally integrate everyone in every way — data, voice, image, video.
Of course, as the Internet’s business value increases, so does the need to protect the systems composing it and the business processes depending on it from attack. This is especially true as fewer human intermediaries are directly in the business process loop due to the increasingly direct system-to-system operation enabled by Web services. Some of the security challenges magnified by the Internet are listed in Figure 1.

As a result, while use of the Internet has been a key goal, integration and security have topped the yearly lists of CIOs’ most difficult challenges. Integration, especially across the Internet, creates too much complexity and too little reliability — and trust — because too many diverse systems, accessed by too many external users, are being integrated across too many organizational boundaries, with too many incompatible integration and security services.

Thus, the need to simplify and improve the Internet’s diverse existing integration, management, and security services is obvious. Given that the Internet’s fundamental purpose is to create the network effect by enabling integration across administrative and trust boundaries, traditional security architectures, which emphasize isolation, perimeter defenses, and centralization, must be rethought.

While the need is obvious, the solution is subtle. We do not need yet another standalone security architecture. We need an architecture for interoperating across standalone security infrastructures. We can no longer afford (if we ever could) to rip and replace multiple existing security services with one monolithic service set. We must find a way to layer over existing technologies — without introducing yet another traditional security technology. Traditional security infrastructures are simply not designed to interoperate with alternative (much less competitive) security technologies — they are designed to replace them (e.g., replacing distributed computing environments [DCEs] with Kerberos).

The solution is to overlay existing security technologies with a service-oriented architecture (SOA). The essential goal of an SOA is to enable general-purpose interoperability among existing technologies and extensibility to future purposes and architectures. The goal of an SOA is not to design any new functionality (e.g.,

---

**Increasing Security Requirements**

- Increasing co-opetition means more:
  - Consortia and joint ventures with competitors
  - Information sharing across supply chains
  - Security shades of gray
- Increasingly dangerous attacks mean more:
  - Sophisticated attacks
  - Persistent attackers
  - Sharing of exploits
  - Attackers beyond legal reach
- Fewer people in the loop mean more:
  - Rapid exploitation of breaches
  - Less time to react
- Increasingly mobile agents mean more:
  - Users roaming across administrative domains
  - Software roaming from servers to clients
- Increasingly dynamic systems mean more:
  - Frequent software updates
  - Frequent user updates
  - Frequent provisioning, configuration, and integration changes

---

Figure 1
a new encryption design, a new design for transmitting data across a medium, a new authentication method); instead, it is to enable interoperability across what is already available and to ensure such interoperability can be extended to include future innovations. Simply put, an SOA is an architecture, inspired by the Internet and the Web, for enabling extensible interoperability based on network concepts such as protocols and intermediaries.

However, even though an SOA may be a solution, it is not a panacea. Although it suggests a solution to the interoperability challenges raised by existing technologies, it does not solve any of their other problems (e.g., manageability, performance, ease of use).

The Invention of the Service-Oriented Architecture

The Internet's protean nature is no accident. Its designers architected it essentially for interoperability and extensibility:

*The top level goal for the DARPA Internet Architecture was to develop an effective technique for multiplexed utilization of existing interconnected networks.* — The Design Philosophy of the DARPA Internet Protocols [emphasis added].

*Heterogeneity is inevitable and must be supported by design. Multiple types of hardware must be allowed for ... Multiple types of application protocol must be allowed for, ranging from the simplest such as remote login up to the most complex such as distributed databases.* — Architectural Principles of the Internet, RFC 1958.

The designers did this by designing the Internet to be an open network architecture that is both technology-independent and application blind (The Flexible Specialization Path of the Internet — emphasis added). This Internet concept has gone by many names over the years: catenet, meta-network, virtual network, network of networks, spanning...
layer, and now SOA. It is the Internet’s SOA that has been the key to its adaptability, cost-effectiveness, and efficiency (see Figure 2).

Thus, the Internet uses generic IFaPs (IP address, IP packet, and TCP/IP protocols) to virtualize a diverse array of specific concrete communications services (e.g., Ethernet, Token Ring, Frame Relay, SONET, PSTN, Wi-Fi) into a single federated communications Web. The Internet’s meta-network approach of leveraging existing networks instead of replacing them is one of the primary reasons why the Internet succeeded in creating a network effect that displaced all other proprietary WANs.

An SOA is best visualized as an hourglass-shaped spanning layer, because a spanning layer resembles the narrow waist of an hourglass — enabling a wide variety of applications above and virtualizing a wide variety of technologies below (see Figure 3).

The Web extends the spanning-layer concept from the Internet’s communications layer to create an application-level spanning layer. Inspired by many of the design principles that shaped the Internet, Tim Berners-Lee based the Web’s SOA on just three fundamental IFaP standards:

- URL (identifier)
- HTML/MIME (format)
- HTTP (protocol)

The Web’s SOA has enabled it to achieve the network effect as well.

Evolving Beyond the Limits of the Web to Web Services

Although the Web’s IFaPs have proven to be amazingly technology-independent — spanning virtually every underlying technology, from J2EE and .Net to CICS and AS/400 — they have not proven to be equally application blind. Up until now,
the Web has been primarily applied to integrate people to services via document-centric browsers — creating the document Web.

Web services extend the boundary-spanning integration capabilities of the document Web by using a fully service-oriented architecture to define a business services Web enabling simple, uniform, and direct integration among services across the Internet. But creating a basic Web services architecture is not enough. To achieve the ultimate goal of a trusted business services Web, traditional security architectures must be rethought to adapt to Web services’ SOA.

Concept of a Service-Oriented Security Architecture

How then should security be redefined by an SOA to create a Service-Oriented Security Architecture (SOSA) reinforcing the network effect of the Internet, the Web, and Web services instead of impeding it? The key is to ensure that the core standards (IFaPs) deliver the essential characteristics of an SOA (see Figure 2).

With regard to the SOA’s federation requirement, traditional security services are where network services were when the Internet was invented 30 years ago — each new security standard still seeks to replace what is in place. But there is often little need for new security standards for the basic services (authentication, authorization, encryption, digital certificate, and digital signature), given the myriad ones that have already been deployed across organizations.

Instead, what is sorely needed is a security spanning layer enabling all these existing security services to interoperate. This is the fundamental goal of the emerging XML and Web services security standards — not to replace the diverse security infrastructures already deployed, but to enable them to interoperate end-to-end.3

What then should define the core IFaPs that shape the SOSA spanning layer? The answer is trust. The fundamental underpinning of any SOA is trust:

Trust is the characteristic that one entity is willing to rely upon a second entity to execute a set of actions and/or to make [a] set of assertions about a set of subjects and/or scopes. — OASIS WS Security (Core).

For a service consumer to delegate responsibility for executing a process to a service provider, a policy of trust must underlie that delegation. For example, before delegating responsibility for holding her savings to a bank, a consumer must first trust the bank. Thus, the ultimate goal of a SOSA is to enable such trust by establishing and enforcing security policies defined by a trust model, without regard to specific enforcement mechanisms. Accordingly, the core IFaPs of a SOSA should not be based on any specific enforcement mechanism, but should be based on a generic model of policy-based trust:

3 Of course, the goal is not to permanently embed existing security infrastructures, especially almost obsolete ones. Rather, the goal is to not require their immediate replacement in order to adopt the SOSA and instead leave the issue of their eventual replacement as a separate business issue.
The Intersection of Web Services and Security Management:
A Service-Oriented Security Architecture

- **Identifiers** for consumers and services
- **Formats** for tokens and policies
- **Protocols** for exchanging tokens and policies

In other words, the SOSA should be defined in terms of *generic protocols* for exchanging *generic (token) claims* offered by service consumers and *generic policy rules* regarding such claims enforced by service providers. The advantage of this approach is that, in a highly federated scenario, only such generic policy and token metadata need be generically exchanged across diverse administrative domains, leaving such domains free to use different concrete policy and token mechanisms (e.g., Public Key Infrastructure — PKI, Active Directory, Kerberos).

The interoperable **formats** of an SOSA should define a generic policy model, modularized into the following conceptual building blocks:

- **Policy**: A set of policy assertions
- **Policy Assertion**: A specific preference, requirement, capability, or other property
- **Security Token**: A set of security claims
- **Claim**: A policy-relevant statement made by or about a subject accessing a service

The interoperable **protocols** of an SOSA should define a generic process model, modularized into the following subprocesses:

- Generating and distributing service security policies via service definitions (e.g., WSDL) and service directories (e.g., UDDI)
- Generating and distributing security tokens
- Presenting tokens in messages and service requests
- Evaluating whether presented tokens meet required policies

Finally, the interoperable **identifiers** of an SOSA should define a generic reference model for subjects, resources, services, and policies. URIs fit this bill quite nicely.

Of course, such SOSA IFaPs must embody some concept of federation via intermediaries. Competing and complementary influences have produced roughly similar concepts of federation embodying roughly similar policy processing concepts. These have been based on distinctions among the following types of intermediaries:

---

4 A more general and perhaps clearer definition of policy assertion: *A rule that governs a choice in the behavior of a system*. A Survey of Policy Specification Approaches.

5 Security tokens can be either binary (e.g., X.509, Kerberos ticket) or XML (e.g., SAML, XrML). A certificate (e.g., X.509) or a ticket (e.g., Kerberos) is simply a *signed* security token. Note that XML tokens can be signed as well; thus, certificates are also either binary or XML.
The Intersection of Web Services and Security Management: 
A Service-Oriented Security Architecture

- **Policy Enforcement Point**: A service that enforces policy decisions, typically at the point of access (e.g., firewall, OS, application)
- **Policy Decision Point**: A service that applies tokens (claims) to policies to generate a policy decision
- **Policy Administration Point**: A service that generates policies
- **Security Token Service (Authority)**: A service that issues, validates, and exchanges tokens (claims)

Several standards embodying various aspects of such a SOSA have been designed over the past several years (see Figure 4).

**Definition of a Web Services Security Model**

Nowhere is this SOSA rethinking of traditional security architectures (e.g., isolation, perimeter defenses, and centralization) more evident than in the evolving Web Services Security Model (WSSM). Although Web services are part of the security problem, because they represent immature technology for enabling system-to-system interoperation across boundaries, they are also part of the solution, because they can be used to implement a solution to the security issues raised by such interoperation. In this way, Web services are just like the Internet — creating security challenges and enabling their solution. For example, despite the fact that TCP/IP is unsecured, it nevertheless provides the foundation for Web single-sign-on solutions. The key is to build up a secure infrastructure out of partially secure layers.

A number of vendors are driving the effort to define the WSSM, which embodies the SOSA outlined above (see Figure 2). The various standards that define the WSSM can be roughly organized into three subarchitectures (see Figure 5):

- **Trusted Communication**: SOAP, WS-Security, WS-SecureConversation
- **Trust Management Web**: WS-Trust, WS-Federation

---

6 For an extensive list of Authentication and Authorization initiatives, see Authentication and Authorisation: Report to Sparta.
8 WS-Authorization and WS-Privacy are not yet published.
Since Web services are based on a model of SOAP-message-style invocation of a WSDL service interface, the first two subarchitectures are defined as extensions to SOAP and WSDL to enable the exchange of security claims and policies as part of the invocation of a Web service. The third subarchitecture is composed of a set of WSDL interfaces defining a security token service, which enables a trust management Web enabling federated trust.

**Trusted Communication**

The WS-Security standard defines a trusted message-spanning layer:

> WS-Security is flexible and is designed to be used as the basis for the construction of a wide variety of security models including PKI, Kerberos, and SSL. Specifically WS-Security provides support for multiple security tokens, multiple trust domains, multiple signature formats, and multiple encryption technologies. — WS-Security.

WS-Security only defines an extensible message header model for applying existing digital signature, encryption, and security token mechanisms to the individual elements in a SOAP envelope (i.e., header and body blocks), so that the recipient can trust the contents of the message and its sender. It does not define any new such mechanisms: This specification is intended to provide a flexible set of mechanisms that can be used to construct a range of security protocols; in other words this specification intentionally does not describe explicit fixed security protocols. — WS-Security.

WS-SecureConversation simply enables two endpoints to reuse a shared security context for a message conversation consisting of multiple SOAP message exchanges using a series of derived keys to increase security.

---

9 OASIS has more accurately renamed its current revision of WS-Security “SOAP Message Security.”
The Intersection of Web Services and Security Management: A Service-Oriented Security Architecture

Trusted Service

The WSSM policy-based standards enable the creation of extensible policy definitions and their attachment to Web services interfaces. The following standards define this policy model:

- **WS-Policy**: Defines basic policy, policy statement, and policy assertion models
- **WS-Policy Assertions**: Defines a few generic policy assertions
- **WS-Policy Attachment**: Defines how to associate a policy with a service, either by directly embedding it in the WSDL definition or by indirectly associating it through UDDI
- **WS-SecurityPolicy**: Defines security policy assertions corresponding to the security claims defined by WS-Security: message integrity assertion, message confidentiality assertion, and message security token assertion

WS-Policy is an extensible model for expressing all types of domain-specific policy models: transport-level security, resource usage policy, even end-to-end business-process level policy. Of course, the only policy assertions standardized so far are those defined in WS-SecurityPolicy and WS-Policy Assertions. WS-Policy is also able to incorporate other policy models such as SAML and XACML.

Trust Management Web

While the Trusted Communication and Trusted Service subarchitectures enable a service consumer and provider to directly interoperate in a secure and trusted manner, both assume that the same underlying security technology (e.g., identical username/password mechanisms) is used at both endpoints. Furthermore, they assume that both endpoints are encompassed by a single sphere of trust\(^{10}\) (e.g., both sites trust the same certificate authority). This means that these specifications alone are insufficient to define how to send a message from one sphere of trust using one kind of key technology (e.g., Kerberos) to a different sphere of trust using a different key technology (e.g., PKI). Such federated end-to-end security interaction will be enabled by the Trust Management Web.

To enable such federated end-to-end interaction requires the specification of a trust authority that acts as an intermediary (a.k.a. broker or gateway) between spheres of trust. Such an intermediary brokers or gateways between two spheres of trust by transforming security tokens in one sphere of trust to security tokens with an equivalent degree of trust in the other sphere of trust.

**The First Step: WS-Trust**

Although significant portions of the WSSM are still unpublished (e.g., WS-Federation, WS-Privacy), WS-Trust takes the first step by conceptually defining

---

\(^{10}\) Sometimes referred to as trust domains.
such an intermediary, called a **Security Token Service** (STS — see Figure 6). As its name implies, an STS simply issues, validates, and exchanges various types of security tokens in response to requests. WS-Trust defines several WSDL interfaces for requesting and receiving such security tokens.

An STS is very much like a PKI certificate authority (CA). However, it goes beyond the traditional concept of CA in two important ways. First, the STS issues all types of tokens (e.g., authorizations, attributes, roles), not just identity tokens. Second, the STS does not simply issue and validate tokens — it **exchanges** them. The key to an STS’s brokering role in a federated trust management Web is its ability to exchange one type of token (e.g., X.509 certificate) for another (e.g., Kerberos ticket). Of course, to carry out such an exchange, an STS must be trusted to issue certain types of tokens, and it must trust other types of tokens issued by other STSs.

For example, imagine an analyst registering for a trade show. The analyst requests an attendee badge at the registration desk (the STS) by providing her driver’s license, passport, or other government-issued photo IDs (i.e., external identity tokens) and a business card showing that she is indeed an analyst (i.e., an external role token). The registration desk issues an attendee badge (i.e., an internal identity and roles token) to the analyst with colors and labels indicating claims regarding the analyst’s role (vs. roles such as press, exhibitor, speaker), which enables access to the sessions and the show floor. At
The Intersection of Web Services and Security Management: 
A Service-Oriented Security Architecture

each session room (i.e., a service), the badge is checked to ensure that the analyst’s role is authorized to attend sessions (i.e., a policy enforcement point). In this example, the registration desk is federating between the external business and governmental spheres of trust and the internal sphere of trust of the show.

Thus, the role of the STS is to bridge existing isolated security islands into federated security island chains through the exchange of diverse types of security tokens. These security islands could be a customer and its suppliers, or they could be a set of business units formed by a series of mergers and acquisitions. Each island would maintain its own set of interior security protocols and administration, but each would also establish a set of border security protocols and administration, which determines the other islands in the island chain it trusts and in what ways it trusts them.

Or think of such federation as a WAN that links LANs via gateway routers. Kerberos is like a security LAN — deployed primarily within a single administrative domain. Enterprise PKI via a corporate certificate authority is another security LAN. Then STSs act as WAN gateway routers federating these two LANs. It is no coincidence that such distinctions sound similar to the Internet border gateway protocol vs. interior gateway protocol distinction used in routing packets across the autonomous systems that make up the Internet. The same conceptual design issues concerning federation are going on in both cases.

In the case of routing packets, the concern is how to create a unified end-to-end routing path that dynamically spans independently routed networks. In the case of dynamically federating trust, the concern is how to create a unified end-to-end chain of trust that spans independently secured networks. Thus, the WSSM reinforces the view that end-to-end security is a network routing concept. Just as the Internet enables diverse data communications networks to interoperate via common packet routers, and Web services enable diverse XML-processing networks to interoperate via common SOAP intermediaries, the WSSM enables diverse trust-enforcement domains to interoperate via common token routers (a.k.a. STSs). This enables a trusted business services Web to be layered over the Internet’s trusted communications Web.

However, just as a network of routers must be managed by the Internet’s management network to provide the transparent Internet to users, so too the network of STSs must be managed to provide the transparent trusted business services Web. We call this network of STSs the Trust Management Web. And just as the Internet’s management network is composed of distinct functional networks, such as the network of DNS servers providing host name services, or the network of SNMP servers providing management services, the Trust Management Web may be composed of distinct sets of STSs, each offering a different trust-enabling service (e.g., authentication servers enabling federated identity, authorization servers enabling federated authorization).

---

13 Domain Name System.
The Road Ahead: Filling in the WSSM

While WS-Trust provides a detailed glimpse into the basic architecture of the Trust Management Web, it gives no insight into the federated administration issues raised by such an architecture. For example, WS-Trust leaves completely open the complex issue of trust delegation topologies among STSs. In fact, WS-Trust refers to several possible delegation models: fixed trust roots, trust hierarchies, and authentication service. While ensuring that the WSSM is general enough to accommodate all such topologies is good design, a concrete implementation of the standard ultimately must make choices. Presumably, WS-Federation will provide such concrete guidance.

Then of course there is the entire set of issues involved in managing the comprehensive life-cycle spanning the components that make up the WSSM (e.g., the STSs, the end policy attachments, the underlying security mechanisms). For example, currently there is no standard for discovering services that require security management, nor services that can provide such management. However, the draft specification, WS-Management, is expected to deal with some of these issues in a way that is compatible with UDDI’s role as the Universal Description, Discovery and Integration of Web Services. At the very least, the WSSM needs to define a standard for provisioning security tokens and policies across STS intermediaries and Web services endpoints. Currently, the OASIS SPML (Service Provisioning Markup Language) is the leading standard addressing such issues and is expected to be used in support of the WSSM.

Furthermore, the WSSM needs a standard mechanism for managing the “soft state” of distributed policies and tokens. Any globally scalable distributed system must cache a certain amount of metadata within local administrative domains. Thus, policy changes in one part of the federated security network may not be propagated to other parts of the network immediately. Accordingly, there must be some mechanism for ensuring the eventual consistency of such information, even in the face of occasional disconnects among regions (a.k.a. network partitions). DNS is perhaps the paradigm example of an architecture for dealing with this issue.

Engineering a highly consistent and available implementation of WSSM that can tolerate sporadic network partitioning will be a task left to vendor competition. An interesting approach to formalizing such an architecture is BASE (Basically Available, soft-State, Eventual consistency), an alternative to traditional ACID approaches to distributed consistency that is based on the concept of continually attempting to recover consistency instead of simply preventing inconsistency.16

A final missing piece of the puzzle is a well-defined model for end-to-end audit, logging, and reporting of distributed service requests. Given that WSSM enables attribute-based authorization, a single identity token may not be used end-to-end.

---

15 For a comprehensive discussion of such trust delegation issues, see CA-CA Interoperability by the PKI Forum, now part of OASIS.
16 Extensible Cluster-Based Scalable Network Services (1997).
Accordingly, it is essential that an authorization decision be traceable, retroactively, to the true identity of the entity accessing the service.

One way to enable such end-to-end auditability is to include the business-level identifying metadata in each SOAP header so that each SOAP agent (intermediary) can capture that metadata in its audit logs. Then tracing identity end-to-end would consist of tracing through each SOAP node’s audit log to discover each identity transformation (e.g., from John Doe to Yale student). Again, such traceability/auditability capabilities will be where vendors differentiate their implementations of the WSSM, though interoperability standards may ensure that each vendor can coordinate auditing responsibility across administrative domains.

**Comparison to Traditional Security Architectures**

Given that the WSSM is a Service-Oriented Security Architecture, it is explicitly designed with traditional security architectures in mind. In fact, the WSSM is an attempt to generalize the essential features of the mainstream security architectures so that it can provide interoperability among them. The uniqueness of the WSSM is this essential focus on interoperability across diverse security architectures. It does not define any concrete signature or encryption mechanisms because it expects to leverage existing ones. And although it does define some of its own security policy assertions and security token claims, it is designed to primarily leverage other token languages, such as SAML and XrML.

As discussed above, because the WSSM is an SOSA — focused solely on federated interoperability — it is not a panacea. It does not address any of the challenges facing existing security mechanisms (e.g., manageability, performance, reliability) other than interoperability. For example, if the WSSM is used to interoperate between a single-sign-on (SSO) product in Organization A and an SSO product in Organization B, the WSSM will do nothing to address the complexity of managing either SSO product. In this regard, the WSSM is no different from the paradigm SOA — the Internet. Using the Internet to interoperate between two LANs does nothing to address each LAN’s manageability, performance, or reliability. Like the Internet architecture, the WSSM is devoted to solving one focused but vital challenge: federated interoperability.

Some security standards take a related generic API (GAPI) approach to interoperability (e.g., CAPI — Microsoft’s Crypto API; CDSA — Common Data Security Architecture; GSS — Generic Security Services; GAA — Generic Authorization and Access Control). Each of these offers some type of standard GAPI that can be bound to a set of diverse drivers that implement the API.

Although this generic driver approach has been proven by the success of ODBC and device drivers (e.g., printers, video cards), it really only provides a mechanism for portability, not federated interoperability. Although a GAPI makes the underlying service implementation transparent to the application developer, it only enables binding to a single underlying service implementation at a time, not a federated network of diverse service implementations. For example, ODBC enables a client application to bind to any one of a set of diverse databases, but it
does not enable *all the* diverse databases to be used as one federated database (e.g., joining across the databases).

The WSSM is less concerned with a universal GAPI, though it is not incompatible with any of them. The WSSM is primarily concerned with a universal set of IFaPs for interoperating among diverse security domains. Thus, it is more like the Internet-routing IFaPs, and GAPIs are more like dynamically bindable TCP/IP stacks for laptops, which can be bound to a dialup, LAN, or Wi-Fi connection.

Other security standards enable federated administration (e.g., Kerberos realms, Active Directory Forests, cross-certifying PKI CAs), but each requires that all administrative domains implement the same technology (e.g., all running Kerberos or all running PKI). WSSM, on the contrary, does not specify any unique security technology. It is simply defined in terms of how it binds to existing standards.

Two competing standards that attempt to deal with not only federated administration but also federated identity across diverse identity services are Project Liberty and Microsoft’s Passport. Microsoft has already announced that future versions of Passport will be based on the WSSM. Thus, the only issue is whether Liberty will adopt the WSSM as well. Given that Liberty is working closely with OASIS on SAML and other aspects of the Liberty architecture, and given that key standards underlying the WSSM have been turned over to the OASIS Web Services Security technical committee, we may see eventual convergence between the two standards under the umbrella of the WSSM.

Finally, for an interesting example of an SOSA based on SAML, which got underway before the WSSM was sketched out, but is very compatible with it, see the Shibboleth Project at shibboleth.internet2.edu. Shibboleth describes itself as an “Interrealm Attribute-based Authorization for Web Services.” About 20 universities and other institutions are currently piloting Shibboleth for Web Single Sign-On across institutional boundaries.17

### Impact on Security Vendors and Markets

As with IP, Web, and other interoperability standards (e.g., SCSI), once interoperability ensures that all products are roughly identical functionally, this leads to rapid commoditization of the underlying technology components. One can see this in the IP network component market, the Web server and browser markets, and even the SAN/SCSI storage market.

However, in such a commoditizing market, vendors shift the competition from the more and more generic components to focus on unique solutions for dealing with the complexity that inevitably arises when large numbers of commodity components are federated to provide a solution. Thus, vendor solutions shift to focus on ease of deployment, QoS, and ease of management.

Accordingly, even if the WSSM were to completely standardize interoperable secure Web services across organizational boundaries and across commoditized

---

17 Note also that the Open Grid Services Architecture (OGSA) is adopting the WSSM. See the OGSA Security Roadmap.
security mechanisms, this will simply lead to an upsurge in products and services for managing the life cycle of the commodity components constituting the WSSM. Vendors well positioned to benefit from the commoditizing interoperability enabled by WSSM are those that are already grappling with the management issues surrounding complex and diverse security architectures.

**Bottom Line**

The capabilities for interoperability and integration provided by the Internet, the Web, and Web services are enabling the evolution of the service economy toward more and more federated business models on a global scale. However, one of the biggest drags on the pace of such evolution is the lack of a security architecture that enables trust relationships among business entities to be established and administered in a more standardized and more automated way.

For the emerging Web services architecture to succeed in extending the remarkably successful *document Web* into a *trusted business services Web* that reliably spans the globe, its designers must apply *service-oriented security architecture* (SOSA) concepts to architect a new *trust management Web* that can be federated across administrative domains and spheres of trust. The fundamental goal of such a SOSA is not to replace the diverse security infrastructures already deployed, but to enable them to interoperate end-to-end. The technology of such SOSA concepts will be realized by the evolving Web Services Security Model.

However, the ability of businesses to trust such technology will not be based on the generic technology alone. To bootstrap into trusting *technology* for delegating trust, business must first trust *strategic vendors and partners* to apply such technology to specific business relationships. Such vendors will be trusted only when they have earned that trust by repeatedly delivering successful business solutions. Part of such trust should be based on the vendors’ mastery of the diverse technologies to be federated by the WSSM and its competence and capabilities in deploying, managing, and auditing such technologies. It is from this initial direct personal relationship of trust with a few key partners that the indirect automated Web of trust will be woven to span the globe.

*Nick Gall is a senior vice president with Integration & Development Strategies, a META Group advisory service. Earl Perkins is vice president of Security & Risk Strategies, a META Group advisory service. For additional information on this topic or other META Group offerings, contact info@metagroup.com.*
About META Group

Return On Intelligence℠

META Group is a leading provider of information technology research, advisory services, and strategic consulting. Delivering objective and actionable guidance, META Group's experienced analysts and consultants are trusted advisors to IT and business executives around the world. Our unique collaborative models and dedicated customer service help clients be more efficient, effective, and timely in their use of IT to achieve their business goals. Visit metagroup.com for more details on our high-value approach.