Entrust is a registered trademark of Entrust, Inc. in the United States and certain other countries. In Canada, Entrust is a registered trademark of Entrust Limited. All Entrust product names are trademarks of Entrust, Inc. or Entrust Limited. All other company and product names are trademarks or registered trademarks of their respective owners.

The material provided in this document is for information purposes only. It is not intended to be advice. ENTRUST DOES NOT WARRANT THE QUALITY, ACCURACY OR COMPLETENESS OF THE INFORMATION CONTAINED IN THIS ARTICLE. SUCH INFORMATION IS PROVIDED "AS IS" WITHOUT ANY REPRESENTATIONS, WARRANTIES AND/OR CONDITIONS OF ANY KIND, WHETHER EXPRESS, IMPLIED, STATUTORY, BY USAGE OF TRADE, OR OTHERWISE, AND ENTRUST SPECIFICALLY DISCLAIMS ANY AND ALL REPRESENTATIONS, WARRANTIES AND/OR CONDITIONS OF MERCHANTABILITY, SATISFACTORY QUALITY, TITLE, NON-INFRINGEMENT, OR FITNESS FOR A SPECIFIC PURPOSE.
# TABLE OF CONTENTS

1. WEB SERVICES AND EDI .............................................................. 3
2. WEB SERVICES AND DISTRIBUTED COMPUTING ...................... 3
3. WEB SERVICES DEPLOYMENT MODEL ........................................ 4
4. WEB SERVICES AND SECURITY .................................................. 5
5. SECURITY POLICY ........................................................................ 5
6. WEB SERVICES IN A SINGLE POLICY DOMAIN ....................... 6
7. WEB SERVICES BETWEEN POLICY DOMAINS ............................ 8
8. SECURITY POLICY MANAGEMENT .............................................. 10
9. SERVICE PROVISIONING .......................................................... 10
10. SERVICE IDENTIFICATION ......................................................... 10
11. ENFORCEMENT ASSURANCE .................................................. 11
12. TRANSACTION AUDIT SERVICES ............................................ 11
13. CONCLUSION ........................................................................... 11
14. ABOUT ENTRUST .................................................................... 11
1. WEB SERVICES AND EDI

Through the 1980s and 1990s large retail and manufacturing companies implemented Electronic Data Interchange (EDI) programs to achieve dramatic reductions in inventory cost and improvements in product quality and customer satisfaction, by eliminating redundant data-entry steps from their business processes. However, the significant non-recurring costs of re-engineering business processes to take advantage of EDI have prevented all but the largest companies from achieving these benefits. This, in turn, has prevented these large companies from fully benefiting, because they also need to expose some of their business interfaces to small and medium-sized enterprises (SMEs) and individuals.

So EDI protocols, on their own, are not sufficient to totally eliminate redundant data-entry steps from a company's business processes. For this reason, the Web was pressed into service so that SMEs and individuals could use the ubiquitous browser and the Internet to interact directly with a large company's business process interfaces.

While this approach addressed the concerns of the large companies, it did not deliver the benefits of EDI to everyone. In addition, the user experience associated with the browser of today can be slow and cumbersome. While the browser experience can be improved through the use of Java applets and ActiveX controls, this approach comes with limitations: unsigned applets restrict client communication to the domain from which the applet was downloaded, thereby placing severe limitations on the business models that can be implemented; and signed applets introduce additional and confusing user dialogues.

The Web services architecture is the new distributed computing architecture designed to address these limitations and unlock the benefits of business process integration on a larger scale.

2. WEB SERVICES AND DISTRIBUTED COMPUTING

The Web services architecture is not the first attempt to define a standard architecture for distributed computing. DCE and CORBA are just two of the previous attempts. Each promised to deliver cost reductions from business process integration within and across enterprise boundaries. And each failed to deliver on that promise. The Web services architecture is simply the current generation. What makes us think that Web services will succeed where previous attempts have failed?

Previous generations of distributed computing demanded a degree of coordination between the developers of services and their clients that is not practical in many enterprise-scale and inter-enterprise applications. Web services promises to solve this through the use of a common language for the messages flowing between end-points, regardless of the computing platforms on which they operate. It also takes a lesson from EDI in adopting a document-centric, as opposed to an object-centric, approach to system design.

Concerns over security and interoperability are two further reasons why previous attempts to solve distributed computing have fallen short of their promise. If an enterprise is going to open up interfaces to its business processes from outside its boundary, it needs strong assurance that the security of its networks and information assets will not be compromised. Furthermore, it is not practical to expect all of one's business partners to choose the same vendor, or even the same distributed computing technology, to expose their process interfaces. Therefore, in order for distributed computing to scale beyond the enterprise boundary, careful attention must be paid to technology-neutrality and multi-vendor interoperability. Previous generations of distributed computing failed in both of these regards.

The basic Web services architecture addresses the security concern primarily through its use of text-based protocols that invoke specific, well-defined, operations. The messages that flow between interfaces are instances of extension languages based on XML and can be examined to ensure that they
are properly-formed. Improperly-formed messages, potentially containing malicious content, can be prevented from entering the internal network.

In addition, the security solution that is being developed for Web services includes a technology-neutral framework, capable of accommodating the full-range of available authentication technologies, such as X.509 and Kerberos. This is a significant departure from previous attempts to standardize security for distributed computing, which required all participants to adopt the same authentication technology.

The interoperability concern is being addressed by the main platform vendors, who have made a significant commitment to achieving multi-vendor interoperability through collaboration in standards bodies and participation in interoperability trials.

3. WEB SERVICES DEPLOYMENT MODEL

So, how does the Web services architecture work? The basic deployment model is shown in Figure 1.

![Figure 1 - Web Services Deployment](image_url)

In this model, the service is developed (1) and the executable code is distributed to the servers on which it is to be hosted (2). In practice, a Web-service may be nothing more than an interface to legacy data or business operations. As a by-product of service development, a definition of the service interface is automatically generated (3). This interface definition is in the form of a Web services Description Language (WSDL) instance, which contains definitions for the various operations and messages that the service provides. To the interface definition is added information about how to communicate with a
specific instance of the service, i.e. what transport-layer protocol and network address to use. The WSDL instance is distributed by means of a registry (4), such as UDDI (Universal Description, Discovery & Integration), so that it can be located and retrieved by the client developer (5). The client development environment automatically generates the stub code needed by the client executable to invoke the service. The client code may be distributed by conventional means or it may be downloaded incrementally on demand (6).

Where the client domain is not a peer domain, for instance in a B2C or B2E setting, the client development may be performed in the service domain, in which case it is the client executable that has to be distributed across the enterprise boundary.

In a variant of this model, the interfaces are defined by an industry association or similar body and both the client stub and service header code can be generated automatically from the standard definition. This variant is more in-keeping with the document-centric approach to system design.

It is interesting to note that, in B2B applications, it is not sufficient for the parties to agree the syntax of the interface, they additionally must agree the meaning of each interface element. This can be achieved either through bilateral agreement or by conformance with an appropriate industry standard.

The messages that flow between client and service are carried in an extensible message format, called SOAP. Intermediaries may exist in the message path for purposes of load-balancing, compression, caching and security operations.

4. WEB SERVICES AND SECURITY

Satisfying the security requirements of the EDI architecture is relatively straightforward. First of all, a policy authority is required to choose suitable security mechanisms. And, in the case of EDI, the Value-Added Network (VAN) serves this purpose. Authenticity and confidentiality of the messages flowing between the VAN and its large customers are commonly provided by leased-line communications. Accountability is achieved by the presence of an independent third-party, in the form of the VAN, and the transaction records that it keeps. Communications over the Web with SMEs and individuals are protected by server-auth SSL and username/password authentication.

Historically, confidentiality of communications has not been a major concern for supply-chain integration applications of EDI. But, for government and healthcare applications it is critical.

Business process integration by means of the Web services architecture can eliminate the common policy authority. In B2C and B2E applications, one party (i.e. the “B” in “B2C”) may act as the policy authority, and the other party may simply conform with the policy chosen by the authority. But, in B2B applications, this may not be appropriate, as each party may have its own security policy governing transactions flowing between them.

5. SECURITY POLICY

In the previous section we introduced the term “security policy”. The Internet security glossary defines security policy to be:

A set of rules and practices that specify or regulate how a system or organization provides security services.

Generally, the term is used to describe a set of plain-language documents that define how the security goals of the organization are to be met through a combination of technical, physical, procedural and personnel safeguards. While the way we use the term here is not inconsistent with this definition, we use
it to mean something more concrete; that is a statement of the actions that must be taken to enforce the controls listed in Table 1.

<table>
<thead>
<tr>
<th>Only authorized end-points can be exposed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only authorized clients can invoke a service.</td>
</tr>
<tr>
<td>Only approved end-point authentication mechanisms are accepted.</td>
</tr>
<tr>
<td>Only approved end-point authentication parameters are accepted.</td>
</tr>
<tr>
<td>End-points do not rely on stale authentication events.</td>
</tr>
<tr>
<td>Only approved confidentiality mechanisms are used.</td>
</tr>
<tr>
<td>Only approved confidentiality parameters are used.</td>
</tr>
<tr>
<td>Only properly-formed messages are admitted.</td>
</tr>
<tr>
<td>Data is released only if authorized by its owner.</td>
</tr>
<tr>
<td>Message elements are properly protected.</td>
</tr>
<tr>
<td>Necessary evidence is collected.</td>
</tr>
<tr>
<td>Necessary commitments are obtained from partners.</td>
</tr>
<tr>
<td>Necessary assurances are obtained from partners.</td>
</tr>
<tr>
<td>Alarms are triggered when required.</td>
</tr>
</tbody>
</table>

Table 1 - Elements of Security Policy

These topics cover all aspects of information security, including: confidentiality, data integrity, system integrity, availability, authentication, authorization, accountability and privacy.

In order to minimize the cost of establishing secure business partnerships, security policy statements must be in a structured form, so that system components can respond to them and automatically enforce and apply the policy, without the intervention of system administrators. Enforcing a policy means rejecting messages that violate the policy, whereas applying a policy means creating messages that will not be rejected on the grounds that they violate the policy. Enforcing policy makes sure that corporate assets are not compromised by policy violation, potentially at the expense of business operations. Whereas, policy application makes sure that business operations are not compromised as a result of policy enforcement.

6. WEB SERVICES IN A SINGLE POLICY DOMAIN

In B2C and B2E applications of the Web services architecture, there may be a single authority that defines the security policy and implements a security architecture to enforce it.
In order to enforce the security policy, the basic deployment model of Figure 1 has to be extended as shown in Figure 2. Security intermediaries are introduced into the message path to protect both the client and the service. And a step is introduced into the deployment process in which the security policy is defined. Policy as it applies to both service requests, responses and fault messages can be distributed in exactly the same way as the interface definition, using WSDL and UDDI. The client security intermediary can then apply the policy to requests and enforce it on response and fault messages, and the service security intermediary can enforce the policy on service requests and apply it to response and fault messages.

In variants of this model, policy development takes place as an integral part of service development and client and service security operations are integral parts of the client and service, respectively. However, building security policy into the application places the responsibility for administering security policy in the hands of the development organization, making it more difficult to modify in the light of changing requirements and a changing threat environment.

In another variant of this model, the client interacts with services from a number of different domains in the course of a single session. There may be no coordination between the various service domains involved, or the service domains may collaborate directly to provide a seamless user experience. This latter approach raises privacy concerns that have to be dealt with through careful system and user interface design.

Security services must be placed carefully in relation to other SOAP intermediaries (see Figure 3). Compression, for instance, must be performed on plain-text messages and, unless care is taken in the system design, load-balancing may interfere with authentication and confidentiality, because a different end-point may be involved in each message exchanged within a session.
Placing security functions close to the application environment allows them to protect against threats in both the external and internal networks. However, sound security practice demands that some security services also be placed at the boundary. For instance, a confidential tunnel can be terminated at the boundary using server-auth SSL. Terminating the confidential tunnel at the boundary ensures that plaintext is available to any other intermediary that requires it.

Authentication and coarse-grained authorization can also be performed at the boundary, using any one of a variety of authentication mechanisms, such as conventional Web-access management techniques or one of the available federated identity solutions. This ensures that messages must pass a rigorous test before being allowed into the internal network. When interfaces must be exposed to unauthenticated clients, messages must be subjected to a different test. In this case, schema-validation is a suitable test.

Schema-validation, fine-grained authorization and other aspects of security policy can be enforced close to the application environment. This allocation of security services also supports an appropriate division of responsibilities between network administrators, who are responsible for the integrity of the internal network and who must have the controls necessary to do that, and application administrators, who are responsible for policy enforcement in the applications and who must have the controls necessary to do that.

Figure 3 – Placement of Security Services

Security intermediaries must be furnished with credentials for authentication and the security services that build on authentication, such as confidentiality, authorization, privacy and accountability. Therefore, solutions are required for managing these credentials.

7. WEB SERVICES BETWEEN POLICY DOMAINS

The situation gets more complicated in B2B applications, where one party cannot simply conform to the security policy dictated by the other party. For one thing, the other party’s policy may not meet the first party’s security requirements. Secondly, if there is more than one partner, whose policy should govern?

Almost all aspects of security policy can only be enforced if all the parties involved implement identical mechanisms and parameter values. This can be a problem where policies have been defined by different
authorities, working independently. It can also be a problem within an enterprise, where systems have been built from a variety of products, each with its own set of security services.

The difficulty could be overcome by requiring that all system components implement all possible security mechanisms, with a broad range of parameter values. This, however, would add unacceptable cost to every system component. Some standard protocols designed to operate across enterprise boundaries acknowledge this problem and provide a way for communicating parties to negotiate a mutually-acceptable set of security parameters within a common mechanism. SSL is an example, in which the cipher-suites available to each party are listed in order of preference and negotiation results in the identification of the single cipher-suite that is most acceptable to both parties. This is then used for securing subsequent communications.

We also see this approach today in the realm of privacy, where the service provider can describe its privacy policy using P3P and the client can describe its policy by means of its browser settings. The policies are combined in the browser and, if they are incompatible, then a warning is raised. In the future, equivalent approaches will be required for other aspects of security policy, including confidentiality, integrity, authentication, authorization and accountability.

The situation is complicated further when each party may choose different and incompatible authentication technologies. This leads to an architecture in which a mapping of security policies has to take place. The external security policy is negotiated with partners, and the internal security policy has to adapt to the requirements of the internal applications, each of which may be different.

One implication of this solution is that security policies must be fine-grained and as liberal as possible, given the application owner’s tolerance for risk. This minimizes the impact of security on interoperability without raising risk to an unacceptable level.

Figure 4 – Multiple Policy-Domain Deployment
Where a service’s client community (or a client’s service community for that matter) is heterogeneous, the security policy applied to each transaction may vary. Therefore, policies must be located, retrieved, combined, cached, applied and enforced automatically at run-time, as illustrated in Figure 4.

The service’s security policy for requests can be distributed as described above, using WSDL and UDDI. The client security policy for response and fault messages can be conveyed in the same way, or as part of the corresponding request. An interface’s security policy may include a public key for privacy purposes and the input parameters to the procedure for verifying signed messages that are sent to the interface (e.g. a trusted-certificate list).

8. SECURITY POLICY MANAGEMENT

In order to minimize the cost and delay associated with establishing a secure relationship between arms-length partners, security policy must be located, retrieved, combined, cached, applied and enforced automatically by system components. Therefore, it must be in a machine-readable form. The syntax of the policy must be such that it can be interpreted as a set of rules for the purpose of enforcement and as a set of instructions for choosing and performing security operations that result in policy-compliant transactions.

At the same time, good governance dictates that security policy be developed, reviewed, approved and audited in accordance with corporate policy. Therefore, it must be possible to produce human-readable versions of the policies that faithfully reflect what is being enforced within the system.

Security policy may change in response to a changing threat environment. And if it changes, then accountability requirements make it necessary to identify the policy that was in effect at the time the transaction in question was processed. Therefore, policy version control and archival is necessary.

9. SERVICE PROVISIONING

Web-service end-points may expose highly sensitive internal resources and functions. Therefore, security policy should include controls over the development process, including approval for the deployment of a Web-service interface, its policies and credentials. Particularly sensitive interfaces may require multiple approval steps to ensure that no individual acting on his or her own can subvert the corporate security policy.

This safeguard can be achieved by means of administrative controls on the issuance of the authorization aspect of security policy, thereby ensuring that access to a service interface is only allowed if it conforms to corporate policy. In this case, any policy that governs access to an interface must have been issued in accordance with the appropriate administrative policy.

10. SERVICE IDENTIFICATION

Service identification and authentication in the conventional Web model builds on the services of Trusted Third Parties (TTPs). The choice of this trust model is dictated by the fact that, in the Web, users are required to authenticate Web-sites operated by companies with whom they have had no previous dealings. It remains the most cost-effective approach for the Web services architecture used in B2C applications.

The situation is different for inter-enterprise business process integration. The limited liability offered by TTPs and the relative ease with which credentials can be exchanged bilaterally will encourage the emergence of different models, leveraging the Web services architecture meta-data distribution mechanism.
11. ENFORCEMENT ASSURANCE

Another area in which new solutions are required involves measuring the effectiveness of a partner’s policy enforcement. Where portions of the business process are out-sourced to service-providers or sensitive customer data are entrusted to partners, the onus remains with the primary customer interface to ensure that its declared policy is adhered to throughout the lifecycle of the data, despite the fact that the primary customer interface has little or no oversight of the partner’s handling of the sensitive data.

In some circumstances, self-certification by a partner is adequate, when accompanied by appropriate guarantees. In other circumstances, certification by an independent third party will be necessary. New standards are required to make these controls cost-effective.

12. TRANSACTION AUDIT SERVICES

Executives in all fields are under increasing pressure from shareholders and regulatory bodies to demonstrate that they are managing the assets of the enterprise in a sound manner. Before technologies for business-process integration were available, humans were involved in all processes that could lead to the disbursement of assets; they represented a natural point of control. From time to time, this control failed. But, at least there was someone identifiable to hold accountable for the failure.

As humans are increasingly removed from business processes, alternative techniques are required to ensure that policy violations are prevented or, at least, detected. This includes alarms and audit analysis tools.

13. CONCLUSION

The extension of business processes beyond the enterprise boundary offers the real promise of productivity improvements to the broad community. However, it also introduces new security requirements, demanding new solutions. The Web services architecture has the potential to provide the necessary solutions in a cost-effective manner. However, careful system design is required to ensure that security requirements are satisfied in a way that has minimal impact on achieving the business objectives. Moreover, several important pieces of the puzzle have yet to be introduced into vendors’ product suites.

14. ABOUT ENTRUST

Entrust, Inc. [Nasdaq:ENTU] is a world leader in securing digital identities and information, enabling businesses and governments to transform the way they conduct online transactions and manage relationships with customers, partners and employees. Entrust's solutions promote a proactive approach to security that provides accountability and privacy to online transactions and information. The company's portfolio of solutions provides security for the broad range of technologies organizations are using today, and planning to use tomorrow including: desktop applications (e-mail, e-forms, files/folders, VPNs and wireless LANs), Web portals, Web services and Identity Management. Over 1,200 enterprises and government agencies in more than 40 countries use Entrust's security solutions, and most recently, both the U.S. Government and the Canadian Government purchased Entrust solutions to secure their network environments. For more information, please visit the Entrust website: www.entrust.com.