Serviam Literature Survey

Part V

Web Service Security

2004-02-19

Anders Toms

SERVIAM-LIT-05

Version 1.2

13 pages

Table of Contents

1 Introduction

Web Services are often described as the technology that will fundamentally change the way business will be carried out over the web by enabling the existing infrastructure to be used for program-to-program communication. This will enable organisations to exchange large amounts of data in an easier and cheaper way than ever before. Among the most commonly mentioned benefits, reduced development time and costs, improved productivity, better and cheaper customer services and increased reusability of code, can be found (Barry, 2003; Clabby, 2002; Fletcher & Waterhouse, 2002).

However, currently, Web Services are not only associated with advantages. There are many more issues that need to be addressed by standard bodies and technology vendors in order for Web Services to become a viable solution for building global service oriented architectures. One such very important issue is security. Many of today's web service implementations are not publicly exposed because of the lack of security that the SOAP version 1.1 specification left. In section 8 of the SOAP 1.1 specification[1] it was simply stated that *"Not described in this document are methods for integrity and privacy protection. Such issues will be addressed more fully in a future version(s) of this document."* Since then (SOAP 1.1 dates back to May 2000), things have happened and a new version (1.2) has been released, which more fully takes security issues into consideration (SOAP v.1.2 Part 1: Messaging framework, section 7). Still, Web Services introduce new security issues that need to be taken into consideration before a web service solution is rolled out. This report will try to clarify what these issues are and provide a picture of what is currently being done when it comes to Web Services security.

1.1 Actors and roles in a Web Service interaction

Depending on how a Web Service is rolled out (e.g. if it is published for public access) there are a number of actors involved in providing and using a Web Service. Naturally, there must be at least one actor that provides the service and another actor that uses the service in order for a service to be consumed. Figure 1 shows the basic Web Services model and the interaction between three different roles involved in the service consumption: service provider, service registry and service requestor (Kreger, 2001). As can be seen in Figure 1, a service provider publishes a service in a service registry (although this is not mandatory). A service requestor (or consumer) searches the registry for a suitable service to use and when one is found, the service requestor binds to the actual service. Both the provider and the consumer are logical constructs meaning that a service can exhibit characteristics of both (Kreger, 2001).

[1] www.w3.org

13 pages

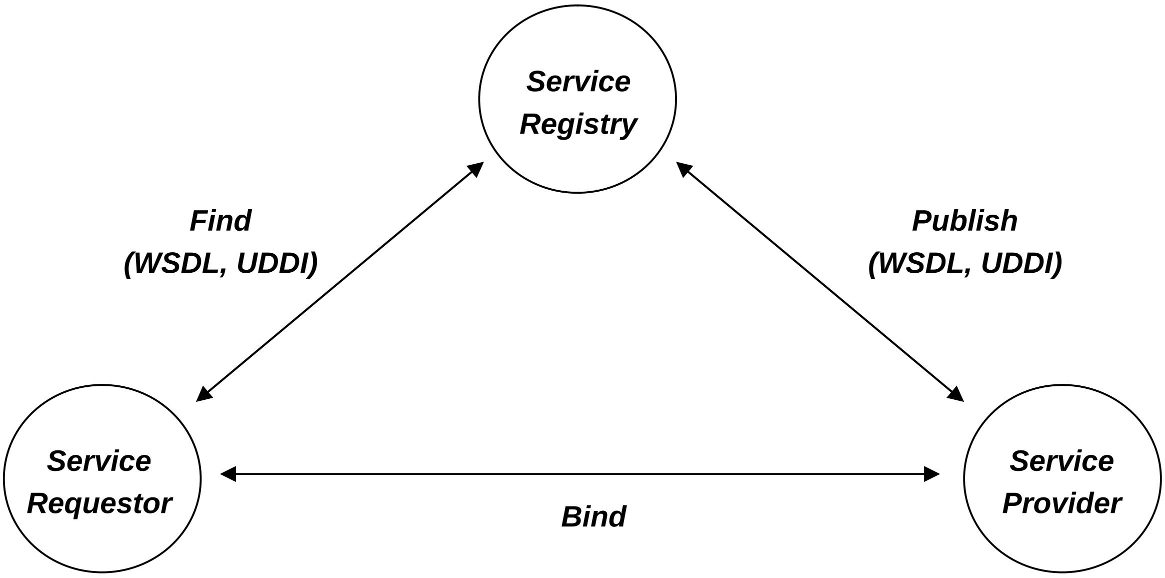

Figure 1. Web Services roles and operations (from Kreger, 2001).

It is possible that a service is provided completely internally in an organisation, implicating that all three roles in Figure 1 may be instantiated by the same actor or organisation. Fernandez (2002) identifies four roles involved in the process of providing and consuming a web service, given the fact that the provider of the service may or may not be the same actor as the actual keeper of the service:

- **Provider** – Is a (person or) business that creates a Web Service and places it in a public repository. They only want authorized customers to access their services.
- **Keeper of repositories of Web Services** – These are the institutions that provide public catalogues of services. They must assure authorized access to these catalogues.
- **Keeper of Web Services** – These store the Web Services' code and data. They may be the same as the keepers of repositories or the Web Service providers, but could also be specialized institutions.
- **Consumer of Web Services** – They expect high quality services without malicious software. If they send their data to a Web Service, this data should be used in the proper way and protected from leakage or corruption.

Not included in the above discussion on roles are intermediaries, which constitute a special type of role. Web Services interactions occur between nodes that can either send, receive or both send and receive messages (Newcomer, 2002). This permit interaction to be carried out in the form of a chain of messages being sent between nodes. In such a chain, nodes potentially positioned in between the start- and endpoint, act as intermediaries and provide certain services on the message's way from the requestor to the provider and back again. Thus, an intermediary is a component that sits in between the service provider and the service requestor, forwarding messages and providing value added functionality of some kind (Clark & Irani, 2002). It is not mandatory that only one intermediary lies on the path between the provider and the requestor. On the contrary, there can be several intermediaries on the path and the return path need not be the same as the initial one, thus potentially involving other intermediaries than the ones involved on the request path. Figure 2 illustrates this. It should also be noted that a service provider may, in turn, use other intermediaries as part of a web service implementation back end, which builds a complex network of services and dependencies (Clark & Irani, 2002).

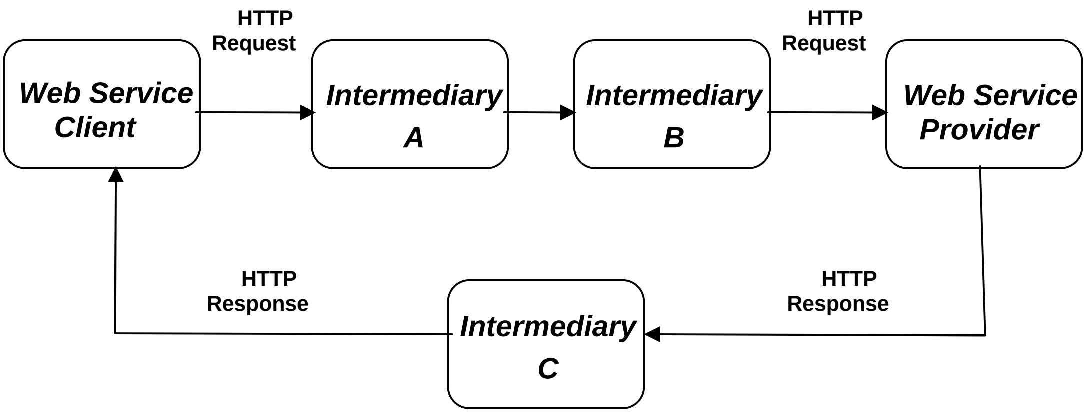

Figure 2. An example Web Service message route (from Clark & Irani, 2002).

In fact, according to Ryman (2003), the complexity of the network and its interactions can grow to the extent that it may even begin to exhibit characteristics of intelligence. After all, Ryman (2003) points out; intelligence may arise from a complex network of relatively simple neurons, as in the neural network our brain constitutes. From a security perspective, it is worth noting that complexity is generally considered a bad thing.

As was mentioned earlier, the intermediary provides certain kinds of services that can be directly related to security, such as authentication and auditing. Clark & Irani (2002) provide the following list of possible intermediary services:

- Authentication services
- Auditing services
- Management services
- Performance improvement services
- Aggregation services

Although an intermediary may provide important services related to security, the intermediary may also introduce a serious threat to the confidentiality, integrity and availability of the messages being sent on its path. The intermediary may for example inspect messages in order to capture or alter sensitive data, effectively performing a so-called man-in-the-middle attack. In order to prevent this from happening, the messages sent across an intermediary should be properly protected.

Protecting the message being sent usually means applying cryptographic techniques of various kinds in order to obscure the data so that an unauthorized interception of a message does not reveal its real content. This ensures the confidentiality requirement (for data in transit) mentioned above and, depending on the cryptographic solution used, may also ensure integrity and other security requirements. Often applied cryptographic techniques include VPN[2] and SSL/TLS[3] that have not exclusively been designed to protect Web Services, but are technologies that carry over from the browser based Internet. Because of this, these techniques have shortcomings that make them useful only as complements to other security techniques (O'Neil, 2002). Using VPN or SSL does not solve all problems related to security in a web service deployment, something that will be discussed later on.

2 Vulnerabilities and threats when applying Web Services

Web Services are essentially an integration technique with a potential of being used for both EAI (internal) and B2B (public) integration solutions. When used for EAI, security issues are sometimes less of a concern since the applications to be integrated are contained and protected inside the corporate network. However, depending on the application, threats may just as well exist internally to the organization as externally, so this situation should not be taken for granted. Nevertheless, a really threatening situation appears when a Web Service that exposes an organization's internal systems is deployed publicly. After all, as King (2003) puts it, when you deploy a web service, *"…you are by default allowing access through port 80 (…) directly in to the heart of you infrastructure."*

The following bullet list present several vulnerabilities, threats and challenges, identified from the literature that the introduction of public Web Services introduces. It should not be considered an exhaustive list though. Some of these threats/vulnerabilities/challenges may impact only one of the roles defined above, while others may impact several. It is difficult, however, to categorize the threats with respect to certain roles since even threats that, at first, look as if they only impact one kind of role, e.g. the service provider, might in some indirect way also impact the requestor. Naturally, these threats/vulnerabilities/challenges are all generic and every specific application can be suspect to more detailed ones, perhaps unique to that particular service implementation. Some of the bullets described below may encompass others.

- **Unauthorized access** (King, 2003) – This is a very broadly defined threat, which may encompass some of the other threats defined below, such as bypassing of firewalls and web application security and possibly also network eavesdropping. It is interesting to note that unauthorized access also includes access to sensitive data at the provider's end of the chain, e.g. by dishonest employees, intruders or similar. Since the data at the provider's site most likely need to be decrypted in order for the provider to perform the necessary operations on it, this may be a threat of great concern. For most applications with only low or moderate security requirements, a trust relationship between the involved parties is often how this problem is solved. However, trust is not always enough. Possible countermeasures to this specific problem are described in Boyens & Gunther (2002), which suggests a special kind of cryptographic technique, called privacy homomorphisms, as a solution to the problem.

- **Parameter manipulation/malicious input** (King, 2003) – Malicious input means that the attacker sends non-expected data as input to the service in order for it to crash and possibly exploit the error state that the service is put in. There are several ways to input

---

[2] Virtual Private Network – A technique in which the protected traffic is encapsulated in a private tunnel over a public infrastructure using cryptographic techniques at the packet level (Bragg et al, 2003).

[3] Secure Sockets Layer/Transport Layer Security – Introduced in 1995, SSL/TLS is the most commonly used security protocol on the Internet. It uses a combination of asymmetric and symmetric encryption in order to protect the data being sent (Bragg et al, 2003).

malicious data to the service, e.g. SQL injection, buffer overflow, parameter manipulation etc. This threat is similar to the one described below, called “Web application security”.

- **Disclosure of configuration data and message replay** (King, 2003) – Message replay is a type of attack in which the attacker captures a legitimate message and later replays that message in order to gain unauthorized access to resources. Depending on how this attack is performed, it could be categorized under the “unauthorized access” threat described above. Disclosure of configuration data, in turn, could possibly be categorized under the “reconnaissance” threat described below.

- **Network eavesdropping** (King, 2003) – Interception of messages sent between the intended parties is always a threat when public infrastructures are used. Traditionally VPN or SSL have been used to protect data in transit. However, these techniques are not always adequate in order to protect a Web Service.

- **Denial of Service** (Burns, 2001) **/ “Domino effect” Denial of Service attack** (DeJesus, 2001) – The complexity that several services coupled in a network may exhibit, as described in the prior section, may also lead to unwanted effects in the case of a “message bombing” aimed at one service in order to perform a DoS attack (DeJesus, 2001). The “message bombing” may have ripple effects that may lead to unforeseen DoS-effects on other, dependent, services. DoS attacks can be very destructive in terms of limiting legitimate access to resources and they can be difficult to completely guard against.

- **Reconnaissance** (Burns, 2001) – Every expert hacker study their target carefully before launching an attack. The attractive feature of Web Services that allows a customer to search for an interesting service to use equally means an attractive feature for a hacker to use in order to gain intelligence about the potential victim. In addition to traditional sources of information like WHOIS databases and DNS servers, UDDI presents an excellent information source for hackers to use.

- **Bypassing of firewalls** (Burns, 2001; O’Neil, 2003; Schneier, 2000) – One of Web Services greatest benefits is also one of the biggest threats. Since Web Services often are implemented using port 80, most firewalls will happily pass the data through without any inspection of the traffic being made. Poorly implemented services can then be exploited to compromise other systems behind the firewall (Burns, 2001). Some vendors already provide firewalls that can handle filtering of SOAP traffic based on target and payload of the message and perform validation against an XML Schema (O’Neill, 2003). The next evolution of firewalls will be to also control what traffic is going out of the corporate network and to create mechanisms that will keep the firewall rules updated on the actual Web Services themselves (O’Neill, 2003).

- **Unintended software interactions** (Burns, 2001) – The complexity of Web Services mean that it will take a while before a sufficiently large body of knowledge exists that can be used for defining industry best practices with regards to security. This is similar to the problem mentioned below.

- **Immaturity of the platform** (Burns, 2001) – Deitel (2002) also points out the fact that security is somewhat empirical in its nature. Certain vulnerabilities will not be discovered until the technology is actually attacked and tested in a real world setting. Using standards and technologies before they are fully developed and tested therefore involves a certain risk.

- **"Business process level vulnerabilities"** (McKenna, 2003) – McKenna (2003) points to the fact that the increased movement of organizations into the world of Web Services will lead to new vulnerabilities at the business process level, in enterprise software like SAP and PeopleSoft. The problem then is that not many people understand the security behind these systems. Most security experts tend to focus on the lower end of the stack.

- **Web application security** (O'Neil, 2003) – Web application security is not a new phenomenon and old attacks, like SQL-injection, directory traversal and URL-attacks, can be tweaked in order to target a Web Service. As an example, the SQL-injection attack can translate into introducing SQL statements in a SOAP message in order to put a back-end database in a vulnerable error state. This threat is similar to the parameter manipulation/malicious input threat described above. All code that implement the service needs to be carefully tested for input vulnerabilities like these before deployment.

- **The challenge of security based on the end user of a Web Service** (O'Neil, 2003) – One problematic aspect of Web Services is that the Web Service does not have "visibility" of the end user since the user interacts with the service through e.g. a web site. In order to be able to make authorization decisions based on end user data, e.g. a username and password, the data needs to be included in the SOAP message sent to the service.

- **The challenge of maintaining security while routing between multiple Web Services** (O'Neil, 2003) – When the path of a SOAP message involves intermediaries, the problem is that encryption techniques that work on lower layers of the stack, like SSL, encrypt the entire communication session without any possibility to selectively choose a specific part of the message. When the message arrives to an intermediate, the intermediate needs to decrypt the data in order to extract information about where to forward the message etc. At that moment, when the data has been decrypted, it is vulnerable to unauthorized access. This lack of possibility to cover the complete security scope is referred to, by O'Neil (2003), as a SOAP-gap. Another problem with a technique like SSL is that it does only provide security for data in transit. When the data is at rest, e.g. stored on a server, it is again vulnerable, i.e. SSL does not provide *persistent security*. The most likely point of attack will be when the data is in decrypted form since it follows the principle of least resistance.

- **The challenge of abstracting security from the underlying network** (O'Neil, 2003) – A Web Service does not have to make use of HTTP for transport. Other protocols, like SMTP, could be used. If this is the case, techniques other than e.g. SSL will have to be used in order to protect the data being sent. When HTTP is used, web server security is often considered the weak link and a likely point of attack.

As is evident from the above discussion, Web Services introduces many new attack vectors and threats that need to be taken into account when a Web Service is deployed. It is not sufficient to only consider technical aspects of network security, but also aspects of administrative security and physical security. If, for example, a dishonest employee gets physical access to the server hosting the Web Service at the provider's site, it may not matter how much resources have been spent on securing the data on its way to the provider. The next section will present emerging standards that will deal with the problems presented here from a technical viewpoint.

3 Emerging Web Services security standards

Since Web Services have become a promising way to deploy new application solutions utilizing existing infrastructure investments and assets, more and more organizations have developed an interest in the technology. And as more and more organizations have joined the Web Services interest group, security has been brought up as one of the most important issues that need to be solved in the near future. Several security standards are under development that, hopefully, will solve the problems brought up in the previous section. The arguably most important ones of these new standards/specifications are presented in this section, together with a short description on security issues concerning UDDI. Just as Linthicum (2004) points out, currently, there are a number of security specifications and standards either on their way or already available that concerns Web Services. In the end, there will not be room for them all.

3.1 XML Encryption

XML Encryption is a specification from the W3C that describes how certain portions of an XML document can be encrypted as well as how any data can be encrypted and presented in XML format (Hartman et al, 2002). Thus, XML Encryption provides confidentiality[4] for Web Services.

The smallest unit of an XML document that can be selectively encrypted is an element. The possibility to encrypt only a certain part of an XML document is helpful in solving "the challenge of maintaining security while routing between multiple Web Services" described in the previous section. An intermediate needs access to certain information in a SOAP message, such as routing information for example, and XML Encryption makes it possible to provide that information in clear text format while encrypting other sensitive information, such as the message payload. Since confidentiality is guaranteed, some of the other problems mentioned in the previous section, like unauthorized access and network eavesdropping, may also be solved.

Selective encryption also has the benefit of being more productive. Encryption is a resource intensive activity and encrypting large amount of data leads to a negative impact on performance. Thus, limiting the encryption process to only the particular data that needs to be protected improves performance. Another important benefit of using XML Encryption is that it provides persistent confidentiality, i.e. the SOAP message is protected not only when in transit, but also when it is in rest. XML Encryption does not define new encryption algorithms or techniques. Instead, existing algorithms like RSA or Triple-DES can be used (O'Neill, 2003).

3.2 XML Signature

XML Signature is similar to XML Encryption in that it explains how portions of an XML document can be digitally signed or how any data can be signed and the signature expressed as XML. Also with XML Signature it is the possibility to selectively sign a certain portion of an XML document that provides the greatest power. XML Signature enables Web Services with a means to achieve integrity, i.e. the ability to detect tampering of the message. Since it is next to impossible to tamperproof a message that is sent over a public network, the next best thing is to detect that a change has occurred during transfer. Hashing algorithms like SHA-1 is used in combination with XML Signature to enable this (O'Neill, 2003). Like XML Encryption, XML Signature is a specification developed by W3C.

[4] Basic concepts like confidentiality, integrity, availability etc is explained in e.g. Mysore (2003).

3.3 SAML

The Security Assertion Markup Language (SAML) is a specification developed by OASIS[5]. According to Hartman et al (2002) OASIS states that their purpose is to create interoperable standards based on XML. OASIS was founded in 1993 under the name SGML Open and is a "*…not for profit, global consortium that drives the development, convergence and adoption of e-business standards.*" (OASIS, 2004).

SAML defines a standard way to transfer security assertions between services expressed in XML format which helps to bridge the gap between different security models (Burns, 2001; King, 2003; O'Neill, 2003). According to Burns (2001) a security assertion is a "statement of fact" that can be tested. The assertion information can be used to perform authentication and authorization decisions and may be inserted into a SOAP message using the WS-Security framework as a guide (O'Neil, 2003). SAML can be used indirectly to provide for authentication and authorization since the information about an earlier authentication event can be included in a SOAP message and passed along the chain of interactions (Newcomer, 2002). Because of the possibility to pass along security assertions that SAML provides, it opens up possibilities for Single-Sign-On (SSO). When system A authorizes an entity based on credentials passed along from system B it means that system A trusts that system B performed the authorization in a correct manner. This is sometimes referred to as *portable trust*. SAML provides a way to solve "the challenge of security based on the end user of a Web Service" outlined in the previous section.

The most recent version of SAML as of this writing is version 1.1. Hartman et al (2002) points out that the previous version (1.0) did not solve certain problems related to the interoperable transfer of security data. Fernandez (2002) also raises certain doubts regarding SAML when he states, "*…this security model seems rather ad hoc and does not follow standard security models, which may result in inconsistencies*". However, he continues by saying that SAML already have been implemented in several security products.

3.4 XACML

XACML stands for XML Access Control Markup Language and is another specification originating from OASIS. XACML standardizes a way to represent access control rules in XML format. This also enables access control policies to be included in a message being sent and that assertion of rights is delivered together with the assertion of identity (Burns, 2003). The two technologies are not linked though, so this is not mandatory. Each technology can be used on its own. XACML currently exist in version 1.0.

3.5 XKMS

XKMS stands for XML Key Management Specification and XKMS 1.0 was first submitted to the W3C in 2001. Currently, version 2.0 is being finalized. The objective of XKMS is to allow an easier way to implement Public Key Infrastructures (PKI) in Web Services (King, 2003). A PKI is complex and difficult to handle which is why XKMS has been developed. XKMS is a Web Service that provides a simplified interface to a PKI that the application can go through. This approach moves the complexity of dealing with the PKI from the application and into the XKMS service itself. The application is then shielded from the underlying complexities (O'Neill, 2003).

[5] Organization for the Advancement of Structured Information Standards (www.oasis-open.org)

3.6 WS-Security

WS-Security can be described as a template for how security tokens are inserted into the header of a SOAP message and for how XML Signature and XML Encryption can be used to protect these tokens. A token is an XML representation of security information, e.g. a claim. A claim, in turn, is a statement saying something about a subject (such as an end user or an entity) that can be used for access control decisions (O’Neill, 2003). WS-Security is thus primarily used for securing SOAP messages and Web Services. Just as Web Services provide a layer of abstraction on top of applications written in different programming languages and deployed on different platforms, so does WS-Security provide a layer of abstraction on top of different security technologies in use in different organisations.

At first, WS-Security may seem to compete with SAML. But SAML and WS-Security are actually not competitors. SAML solves the problem of how to express security assertions in XML format while WS-Security describes how security information is contained in a SOAP message. WS-Security makes use of other technologies like XML Signature and XML Encryption that are suitable for protecting Web Services but are not limited to that. XML Signature and XML Encryption can also be used for other purposes than Web Services security.

WS-Security is only the first of a series of specifications originating from a joint effort between IBM and Microsoft described in an article called “Security in a Web Services World” (IBM & Microsoft, 2002). WS-Security is an important specification because it describes how XML security specifications relate to SOAP and Web Services security. Later specifications will also build on WS-Security. Like SAML and XACML, WS-Security is a specification being developed under OASIS to which it was submitted by Microsoft and IBM in June 2002.

3.7 Security in UDDI

UDDI stands for Universal Description, Discovery and Integration protocol and provides a way for a service requestor (see Figure 1) to search for appropriate services to invoke. The UDDI registry is divided into white, yellow and green pages, similar to an ordinary telephone catalogue. The white pages cover business details while the yellow and green pages cover service details. Two basic types of access are allowed to a UDDI registry; inquire and publish. An inquirer is allowed access to the registry in a read-only form and is not authenticated by default since they are not allowed to publish or alter data. Usually, access to the registry is provided over HTTP since there is no authentication information being exchanged. The publisher is able to insert new records, if properly authorized to do so, and to update its own records. A publisher needs to be authenticated, often using a username and password, which is why HTTPS is commonly used to protect the access to the server.

Besides common security issues, such as host-hardening, authentication, encryption etc, UDDI is also subject to the problems of reconnaissance, discussed in section 3, and trust. The problem associated with trust is that, normally, a business relation is built up over many years and potential business partners are scrutinized in detail before entering a business relationship. UDDI offers the possibility to automate the task of discovering new business partners. However, for this possibility to be used to a greater extent, some kind of certification authority or business brokerage in the form of an intermediate may be needed (O’Neill, 2003).

4 Summary

Getting a grip on security is the next important thing for Web Services if they are to be deployed in a global context and much work is being done in this field at the moment. There exist many different specifications aimed at addressing various security problems related to Web Services. One should keep in mind though, as mentioned earlier, that security is inherently empirical to its nature in that certain vulnerabilities will not be discovered until a particular technology is widely used in real world situations and subject to massive attacks. Web Services, and in particular the proposed standards for securing them, are still very immature which constitutes a big risk. Also, at this stage, most security tools available for Web Services offered by companies like Microsoft, BEA, Sun etc are geared towards programmers and not administrators. This leads to several problems related to hand-coded, non-reusable security solutions (Vaughan, 2003). According to Boubez (Vaughan, 2003) Web Services security should be an administrative task, not a programming one. However, depending on the characteristics of the service, old and proved techniques like SSL can be used to provide a sufficient degree of protection. Still, most services that are up and running today seem to be deployed internally on closed and carefully monitored corporate networks and this will hamper the vision of Web Services as the ubiquitous technique for program to program communication.

Finally, it is unavoidable to mention the scenario given by Berinato (2003) in the article "The future of security" referred to as "the digital Pearl Harbor". Simply put, the digital Pearl Harbor is described as the total break down when everything goes black. After that, those computer experts who survived will take a different approach to application integration; "*After decades spent making access to applications universal, computer scientists and software designers will focus on preventing access.*" (Berinato, 2003). This scenario can be useful to keep in mind as we continue our struggle to interweave systems.

5 References

5.1 Books

Barry, D. (2003) *Web Services and Service Oriented Architectures, the Savvy Managers Guide.* San Francisco, Calif.: Morgan Kaufmann: Elsevier Science. ISBN: 1-55860-906-7.

Bragg, R. Et al. (2003) *Network Security: The Complete Reference.* Emeryville, California: McGraw-Hill/Osborne. ISBN: 0-07-222697-8.

Clabby, J. (2002) *Web Services Explained, Solutions and Applications for the Real World.* Upper Saddle River, NJ: Prentice Hall. ISBN: 0-13-047963-2.

Clark, M. & Irani, R. (2002) Web Services Intermediaries. Fletcher, P. & Waterhouse, M. (ed.) (2002) *Web Services Business Strategies and Architectures.* Chapter 12. UK: Expert Press Ltd. ISBN: 1-59059-179-8.

Deitel, M. (2002) *Web Services, A Technical Introduction*. UK: Prentice-Hall. ISBN: 0-130-46135-0.

Fernandez (2002) Web Services Security. Fletcher, P. & Waterhouse, M. (ed.) (2002) *Web Services Business Strategies and Architectures.* Chapter 17. UK: Expert Press Ltd. ISBN: 1-59059-179-8.

Fletcher, P. & Waterhouse, M. (ed.) (2002) *Web Services Business Strategies and Architectures.* UK: Expert Press Ltd. ISBN: 1-59059-179-8.

Hartman, B, Flinn, D, J, Konstantin, B & Kawamoto, S. (2002) *Mastering Web Services Security.* Indianapolis: Wiley Technology. ISBN: 0-471-26716-3.

Newcomer, E. (2002) *Understanding Web Services: XML, WSDL, SOAP and UDDI*. UK: Addison-Wesley. ISBN: 0-201-75081-3.

O'Neil, M. (2003) *Web Services Security*. US: McGraw-Hill. ISBN: 0-07-222471-1.

5.2 Articles

Berinato, S. (2003) The future of security. *CIO Magazine*, December 30, 2003. Available on the Internet (040127): http://www.computerworld.com/securitytopics/security/story/0,10801,88646,00.html?f=x73

Boyens, C & Gunther, O. Trust Is not Enough: Privacy and Security in ASP and Web Service Environments. Y. Manolopoulos & P. Návrat (eds.): ADBIS 2002, LNCS 2435, pp. 8-22, 2002. Springer Verlag, Berlin.

Burns, S. (2001) Web Services Security – An Overview. Available on the Internet (031202): http://www.sans.org/rr/papers/index.php?id=225

De Jesus, E, X. (2001) Security Implications of Web Services – Web Services need all the security features of any web-based operation, and more. Available on the Internet (031003): http://www.webservicesarchitect.com/content/articles/deJesus01.asp.

IBM & Microsoft. (2002) Security in a Web Services World: A Proposed Architecture and Roadmap. Version 1.0. April 7, 2002. Available on the Internet (031207): http://www-106.ibm.com/developerworks/webservices/library/ws-secmap/

King, S. Threats and Solutions to Web Services Security. *Network Security*, Volume 2003, September 2003.

Kreger, H. (2001) Web Services Conceptual Architecture (WSCA 1.0). IBM Software Group. Available on the Internet (040121): http://www.306.ibm.com/software/solutions/webservices/pdf/WSCA.pdf

Mysore, S. (2003) Securing Web Services – Concepts, Standards, and Requirements. Whitepaper, Sun Microsystems Inc. October 2003. Available on the Internet (031207): http://wwws.sun.com/software/whitepapers/webservices/securing_webservices.pdf

Linthicum, D. S. (2004) WS-Security Standards… What Gives? *Business Integration Journal*, January 2004.

OASIS (2004) Available on the Internet (040127): www.oasis-open.org/who/

O'Neill, M. (2002) Is SSL Enough Protection for Web Services? *EAI Journal*, December 2002. Available on the Internet (040126): http://www.eaijournal.com/PDF/ONeillSSLSecurity.pdf

Ryman, A. (2003) Understanding Web Services. IBM Toronto Lab, July 2003. Available on the Internet (040126): http://www-106.ibm.com/developerworks/websphere/library/techarticles/0307_ryman/ryman.html

Schneier, B. (2000) SOAP, Crypto-Gram Newsletter. June 15, 2000. Available on the Internet (040126): http://www.schneier.com/crypto-gram-0006.html

Vaughan, J. (2003) Q&A: Web Services Security. Available on the Internet (031205): http://www.adtmag.com/article.asp?id=8523