Managing and Securing Web Services with VPNs

Lina ALCHAAL
Netcelo S.A., Echirolles, France
INRIA Rhône-Alpes, Planète project, France
lina.alchaal@inrialpes.fr

Vincent ROCA
INRIA Rhône-Alpes, Planète project, France
vincent.roca@inrialpes.fr

Michel HABERT
Netcelo S.A., Echirolles, France
michel.habert@netcelo.com

Abstract

Web Services constitute a set of technologies that many believe will change the web communication landscape within the next few years. They offer standardized and easy communications for distributed systems over the Internet. However their dynamic and distributed nature requires a well-managed system, and pending security issues prevent their widespread adoption. Meanwhile there is a big rage toward the use of Virtual Private Networks (VPNs) to secure communications in a cost-effective environment like the Internet. In this paper we explain how to merge these two technologies in a new powerful hybrid model that: (1) enables an easy management of web services, (2) provides web services security thanks to the use of dynamic and programmable VPNs, and (3) remains simple and fully integrated.

1. Introduction

Virtual Private Networks (VPNs) [13][12] have become an easy way of securing communications over the Internet. VPNs services are a fundamental part of distributed systems spread over the Internet [10], since they enable the creation of secure data tunnels among remote sites or hosts. In [2] we have introduced a VPN service provider architecture that enables to dynamically build IPsec VPNs between sites or hosts, and which is totally independent from the Internet Service Provider (ISP) used by each of them. This approach is centralized around the Virtual Network Operation Center (VNOC) of the VPN service provider. This VNOC collects all configuration and policy information and remotely configures VPN sites dynamically according to the client requests to join or leave a VPN.

The WWW is more and more used for application-to-application communications. Web services are a new breed of web applications. They are self-contained, self-describing, modular applications that can be published, located and invoked across the Web [14]. Web services perform functions which can be anything from simple requests to complex business processes. Once a web service is deployed, other applications (and other web services) can discover and invoke this service.Web services provide a standard means of interoperability between different applications, running on a variety of platforms and frameworks. One of their strength is the use of standard web technologies [17] (e.g. XML, SOAP, WSDL, UDDI), as will be explained latter on. However other issues like the security and dynamic management of web service architectures are still under discussion.

There is clearly a mutual need between the web service and the VPN technologies. The proposed model fuses several security technologies, IPsec and SSL, and the web services technology into the same melting pot. The resulting hybrid model: (1) enables an easy management of web services, (2) provides web services security thanks to the use of dynamic and programmable VPNs, and (3) remains simple and fully integrated.

This paper is organized as follows: we discuss our VPN approach in section 2; in section 3 we discuss web services and highlight the pending management and security issues; we detail our proposal in section 4 and discuss some design and performance aspects in section 5; finally we introduce related works and we conclude.

2. A Dynamic and Centralized VPN Approach

In this section we introduce our VPN architecture, discuss its features, and quickly compare SSL and IPsec VPNs.

Proceedings of the IEEE International Conference on Web Services (ICWS'04)
0-7695-2167-3/04 $ 20.00 IEEE

2.1. Centralized VPN Architecture

In a previous work [2] we described an IPsec VPN architecture that fully secures communications over the Internet. It relies on a *VPN service provider* which is the responsible of the VPN deployment and management between two or more VPN sites. A VPN site can be a system at a branch office, a laptop or even a system on a trusted partner's network. This centralized VPN approach takes in charge all the security management aspects like the authentication and access control of the sites that want to join the VPN. Everything is dynamic and sites can join or leave a VPN at any time.

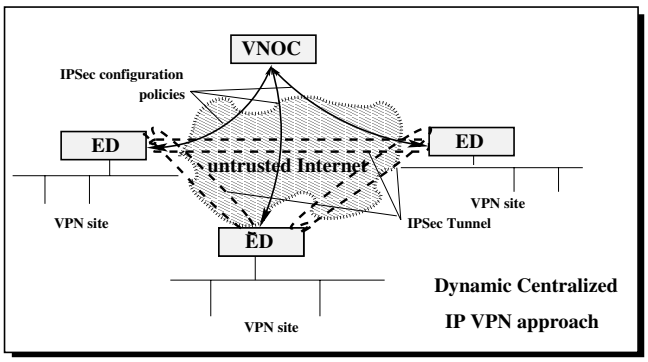

Figure 1. The Dynamic Centralized VPN approach for site-to-site security.

In this approach, a dedicated host, the Virtual Network Operation Center (VNOC), is the central point[1] that distributes policy configurations to the Edge Device (ED) of each site. Those EDs are VPN enabled access gateways, and one of them is necessarily present in each site. Each ED can issue requests to the VNOC to join, leave or query a VPN. For instance, in case of a "join request", if the site is authorized, the VNOC sends back the VPN configuration to the ED of the site and updates the configuration policies of the other EDs concerned by this VPN. This ED/VNOC dialog is based on XML messages sent over HTTPS. Therefore, the site-to-site communications are secured by IPsec [11], the security protocol used in our VPN tunneling approach, while sites-to-VNOC and VNOC-to-sites communications are secured by SSL connections.

This approach has several distinctive features:

- *ISP Independence:* this service is provided by a VPN service provider independent of any ISP. This is a major asset since it does not create any ISP dependency (which can be changed almost transparently) and enables VPN creations between sites connected through different ISPs to the Internet. This is a major asset over other VPN solutions provided by ISPs that heavily rely on their own routing infrastructure.
- *simple centralized approach:* the presence of a centralized VNOC greatly simplifies the configuration, management and possibly billing aspects.
- *relies on well-known building blocks:* security is always done on a point-to-point basis, using well known security protocols like IPsec (site-site) and SSL (site-VNOC).
- *fully dynamic approach:* each ED sends dynamic requests to the VNOC to join/leave a VPN. Those dynamic requests are done on behalf of clients depending on their own needs.
- *many topologies are possible for site-to-site communications:* in this paper we assume that a star topology, centered on the service provider, is created. However this is not compulsory and other schemes are possible for other communication scenarios. For instance [1] introduces the Routed VPN (or VPRN) concept whereby EDs can route packets between various VPN branches.
- *well-managed system:* this approach offers an access mechanism to authenticate and authorize users before they can benefit from the VPN service. It also provides a provisioning tool for the subscription, management, monitoring and billing processes.

Last but not least, this approach has been fully implemented and is commercially available [16].

2.2. Support of SSL-based and IPsec-based VPNs

IPsec and SSL (or the TLS IETF successor of SSL [6]) are two effective ways to provide secure communications over the Internet [22][4] and both can be considered for building VPNs. Yet several fundamental differences exist [5], essentially because IPsec operates at the network layer whereas SSL operates at the application layer. For instance IPsec offers a global security to all the applications, no matter whether they use the TCP or UDP protocol, but it requires that an IPsec stack be installed. On the opposite SSL only offers security to SSL-aware applications, but since most web browsers support HTTPS (i.e. HTTP over SSL), it is largely deployed and used. But from a security point of view, there is no significant difference since SSL and IPsec both support a wide variety of security services.

In practice, building an SSL VPN or an IPsec VPN has more to do with clients needs [3]. Therefore both technologies can be used in a VPN environment and *we consider both IPsec and SSL based VPNs* in our VPN web service architecture proposal.

1 Note that in practice secondary backup VNOCs, located outside of the VPN service provider's network, are used for high reliability purposes.

3. Web Services

We now introduce web services and focus on the limitations of current solutions.

3.1. Web Service Architecture

A typical web service consists of three elements:

- a Service Requester (the client), who requests the execution of a web service,
- a Service Discovery Agency, who typically operates as a repository where a service provider publishes its services, and
- a Service Provider (later referred to as a web service platform), who provides a set of services.

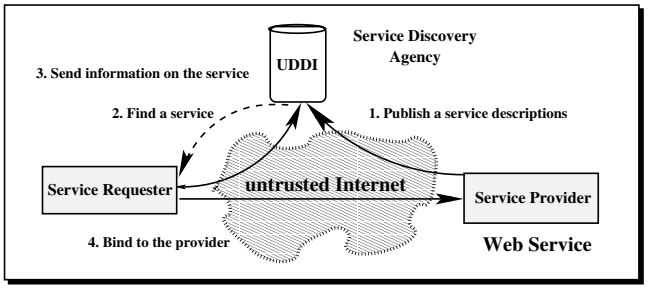

Figure 2. Web Service Architecture.

In a typical scenario, a service provider first describes a web service and publishes it to a service discovery agency. A service requester retrieves the web service description from service discovery agency, and invokes the web service implementation from the service provider.

The use of standard technologies, in particular XML and HTTP, is the main factor of the web service success. In addition to that, a fully functional web service also requires:

- SOAP: It is a specification protocol that defines a uniform way for passing an XML message [15]. A SOAP message contains the name and parameters of the method to call on the service provider, and returns the values to the service requester. SOAP provides a light-weight messaging format that works with any operating system, any programming language, any platform, and which is firewall-friendly.
- UDDI: it looks like a repository service [18] and it offers a mechanism to discover services published by providers. It has two kinds of clients: service providers (who want to publish a service and its usage interfaces), and service requesters (who want to obtain some services).
- WSDL: it is a definition language [23] used to describe where a web service resides and how to invoke it.

The hosting web services platform is typically either J2EE or .NET. Platforms handle runtime issues on behalf of web services, and because of their complexity they need to be carefully managed.

3.2. Evolution of Web Services and Open Challenges

The evolution of web service technologies can be divided into three phases:

- The first phase concerned the basic standards: XML, SOAP, UDDI and WSDL. Thanks to the efforts of such standardization groups as W3C and WS-I (Web Service Interoperability Organization), these standards are now rather mature.
- The second phase addresses the security and reliability issues. This phase is still in an intermediary stage, despite the big efforts of several working groups such as OASIS (Organization for the Advancement of Structured Information Standards), WS-I and others. For instance WS-I is working on critical web services specifications like XML Digital Signature, XML Encryption, HTTP-R, SAML (Security Assertion Markup Language), and XACML (eXtensible Access Control Markup Language).
- The third phase addresses the provisioning, monitoring and system management. This phase is really in the very early stages of discussion, even if it is of utmost practical importance.

Since the functional requirements are essentially covered, challenges have moved to the management and security aspects.

3.2.1. Management Challenges: A successful approach to web service management requires a set of flexible, interoperable security primitives that, through policy and configuration, enable a variety of secure solutions, in addition to a set of policies for deploying, operating and monitoring web services. In this work we concentrate on the external management operations (i.e. the management operations between a web service platform and the client environments) rather than on the internal ones. External management operations include:

- client identification: clients are given a unique identity when placed under management control. Without these identities, monitoring all instances of the web service environment would be complex.
- client authentication and authorization: the goal is to authenticate clients and give them the right or not to invoke services from a web service platform.

- monitoring operations: the goal can be to determine whether a web service instance is working correctly, to get the list of clients currently hosted on the web service platform, the list of web service instances currently responding or not, or the average dispatch time for messages sent between web service instances.
- configuration deployment: configuration policies must be deployed in order to secure communications between a client and a web service platform. By having enough information about clients, generating suitable configuration policies that conform to the clients environments is much easier.

Yet the current web services standards do not handle all above issues. A significant limitation is the management of authorization and security events, as well as the explicit capabilities of monitoring, configuring and identifying.

3.2.2. Security Challenges: Most of the current efforts concern security issues. Many languages are created for this purpose (but are not yet finished). One of these languages, known as Web Services Security Language (WS-Security), defines SOAP extensions that can be used to provide integrity and confidentiality. Other languages use XML extensions for these purposes:

- XML Signature [7] describes how to digitally sign an XML document and provides integrity, signature assurance, and non-repudiation for web data.
- XML Encryption [8] describes how to encrypt an XML document and represents the encrypted content of XML data and the information that enables a recipient to decrypt it.
- XKMS [9] describes an XML-based protocol for delegated trust and specifies protocols for distributing and registering public keys (used in conjunction with XML Signature).
- XACML [20] provide a specification for policies to access XML documents, based on objects (elements to be accessed in the XML document), subject (the user), action (read, write, create, delete)
- SAML [19] describes authentication, attributes, and authorization decision
- SPML [21] Service Provisioning Markup Language for exchanging user, resource, and service provisioning information

These standards do not address the larger problem of how to secure a web service, but how to secure XML data, no matter whether it is part of a web service or not. Moreover they do not address the issue of transport protocol security, although transport is an integral piece of end-to-end secure messaging required.

Other technologies are used for establishing secure connections over HTTP like SSL and TLS. Basically transport security for web service messaging consists in using SOAP over HTTP/SSL or TLS.

Using IPsec is another way to secure web service messages. It provides peer authentication, confidentiality and integrity of IP packets. However fundamental differences exist between SSL/TLS and IPsec as explained in section 2.2.

No matter whether the security technology is for point-to-point or end-to-end communications, a high level of security management is required to prevent conflicts between web service instances and to deploy security policies without any interoperability problem. Additionally, these policies should be generated in a dynamic way, without any human intervention, each time a client requests a web service.

4. Our Approach: VPN Web Services

4.1. Introduction to the VPN Web Service Architecture

The proposed architecture, that merges the web service and VPN technologies, includes all the components of any web service. It is based on a *central Management Operation Point (MOP)* that handles all the management aspects of both the web service and the clients, but is not involved in the web service processing itself. The MOP includes two components:

- a UDDI register where the service descriptions are published, and
- a VNOC where the management of the web service occurs.

In this architecture, we distinguish the *management operations* of the web service platform and client environments, from the *business operations* where clients request the execution of a web service. The MOP is only concerned by management aspects. There is only one management interface in the MOP, shared by all web service platforms and clients, while there might be several business interfaces, one at each web service platform.

In order to unify all operations, a dedicated web service is created between the MOP and each web service platform or client environment for management operations. To avoid confusions, we now distinguish:

- the *management web service*, provided by the MOP to the entities he manages, and
- the *business web services* (or just web services) provided by a web service platform to its clients.

We therefore make a *recursive use of the web service concept* in our architecture. Naturally, WSDL descriptions

are provided for the management and business services. SOAP is used in both cases, and the management functionalities can be discovered in the same way as web services. The traffic within the management web service itself is already secured by SSL as explained in section 2.1.

To simplify management tasks of the various business web services, and to address security requirements, the MOP creates a *dynamic VPN for each business web service*. This VPN is a star VPN that spans:

- the web service platform (center of the star).
- all clients of this business web service: they join (or leave) the VPN dynamically and invoke the web service. They are at the periphery of the star VPN.

From now on we will refer to the business web service created over a VPN by a *VPN web service*.

In this architecture a VPN web service is created by the MOP for each web service offered by a web service platform. Nevertheless a web service platform could use only one VPN web service for all his web services, but this case is not considered in this work because it may require to add AAA functionalities to the platform which contradicts our goals.

4.2. VPN Web Service Phases

In this section we describe more in details the various steps required to set up a VPN Web Service, as well as several types of SOAP messages used to establishing VPN web services. Two phases can be identified during the setup process:

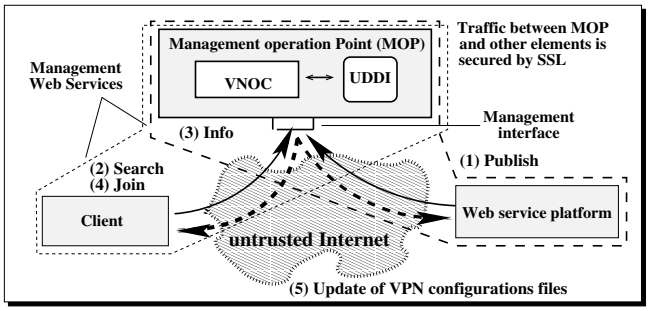

Figure 3. Building Web Service VPN: Signaling Phase.

4.2.1. Signaling Phase and VPN Branch Setup: This phase concerns the management web services, and messages are exchanged between the MOP and the web service platform or the clients. So we can distinguish two kinds of messages:

- The messages exchanged between the MOP and a web services platform: the web service platform first contacts the MOP by sending a *Publish* SOAP message (1), with a WSDL description file attached that contains details of the business interface and the provided services. The SOAP message is processed by the VNOC who verifies the identity and authorization of the web services platform (we assume a list of authorized platforms exists). If OK, the VNOC registers the new service description file into its UDDI register, and associates a new star VPN to the web service platform.
- The messages exchanged between the MOP and a client environment: a client initiates the signaling phase by sending a *Search* SOAP message (2) to the management interface to look for information about a business interface. Once the client is identified by the VNOC, an *Info* SOAP message (3) is returned to the client with the description file of the business interface requested, plus the ID of the associated VPN web service. The client then sends a *Join* SOAP message (4) to the MOP to join the VPN web service. Upon receiving this message, the VNOC adds the client's ID to the VPN members and distributes the new (IPsec or SSL) configuration policies (5) to the client and the web service platform. A VPN tunnel is finally established between them.

Two other types of SOAP messages exist:

- *Leave message*: it is sent by a client who wants to leave one (or more) VPN web services identified by their ID. Upon receiving it, the MOP updates the configuration policies of the corresponding web service platform(s) to delete the VPN tunnels for this client.
- *GetVPN message*: this message enables to get the VPN web services IDs that a client has joined.

All the IDs for the VPN web services, the clients and the web services platforms are generated manually. This manual configuration is due to the need of verifying the requester identities which can be done via traditional ways, like mails or telephone calls and the need of determining the required security level.

4.2.2. Data Transfer Phase: Once the client and web service platform have received VPN configuration policies, the client can then invoke service thanks to the information found in the WSDL description file obtained previously. When the client sends his first SOAP message to the business interface, it invokes the establishment of the tunnel between the client and the business interface. Once established, the tunnel will remain active till the client sends a *Leave* message to the management interface, and this later has updated the VPN configuration policies.

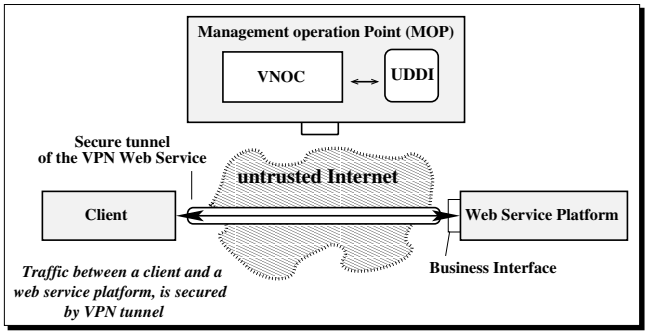

Figure 4. Building Web Service VPN: Data Transfer Phase.

Depending on the security policies deployed in the previous phase, the traffic between the client and the web service platform is secured either by IPsec or by SSL, depending on the configuration policies. The choice of the security protocol(s) depends on the clients security needs when they register to the VPN service. The use of IPsec or SSL will not prevent the use of other technologies such as XML signature or SAML to secure point-to-point communications, but they add another level of security.

Another benefit is that a VPN web service contains only authorized members who are interested in invoking its services. Thus a web service platform does not care about the authentication and access control of clients any more. These security aspects are performed by the VNOC on behalf of the web services platforms.

5. Implementation and Evaluation of a VPN Web Service Platform

In order to evaluate this architecture we implemented a great part of the system. The MOP includes the VNOC (UDDI is missing in this prototype) and uses a J2EE platform to handle runtime issues. The management interface runs in front of a Java Apache server, and the clients use a Perl program to send SOAP messages to the management interface.

We measured the time required to handle management operations on the VNOC, in a real, operational environment. We focused on the three major operations: *Join*, *Leave* and *GetVPN* services Experiments have shown that processing *Join* (figure 5) and *Leave* messages require respectively 978 microseconds and 938 microseconds on average. This is twice as long as the processing of *GetVPN* messages that only require 492 microseconds. This is normal because *Join*/*Leave* operations imply modifications in the VPN database handled by the VNOC.

A problem that affected performance, especially when the number of clients increases, is related to the basic authentication within SOAP messages. We used the SOAP::Lite model to implement the Perl client program. Since SOAP was developed to send authentication information just in case where the server asks for them, the client sends its first SOAP message without any authentication information. This causes the management interface to send back a "401: authentication error" message, and the client must reply with the missing information. We solved this problem by enforcing SOAP messages to always send authentication information in the first message sent to the management interface.

We then measured the time required to set up a VPN web service. This is the time between receiving a *Join* message from a client, and updating the VPN configuration files in the client's side. This time is in fact strongly related to some VNOC parameters that add an additional Δtime to the update process in the current VNOC implementation. In our tests (figure 6) the Δtime ranges between 10 to 20 seconds

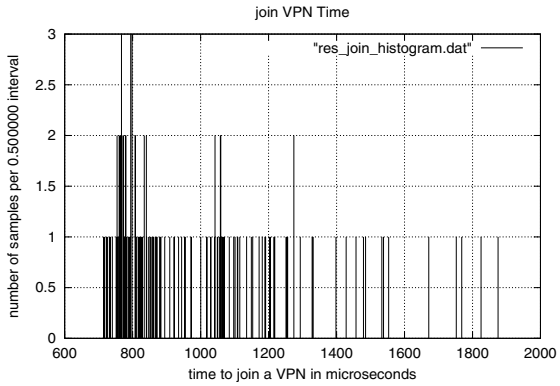

Figure 5. Processing time of a Join message (histogram).

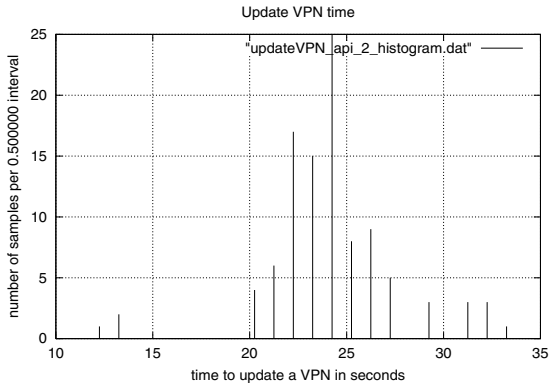

Figure 6. Update time of a VPN web service (histogram).

and we measured an average update time of 24 seconds. This is the waiting time before a client can participate to the VPN web service. It may seem long, but once the client is connected, service invocation is only determined by the web service platform, no longer by the VPN web service framework. Besides we are currently working on this aspect to reduce this initial latency.

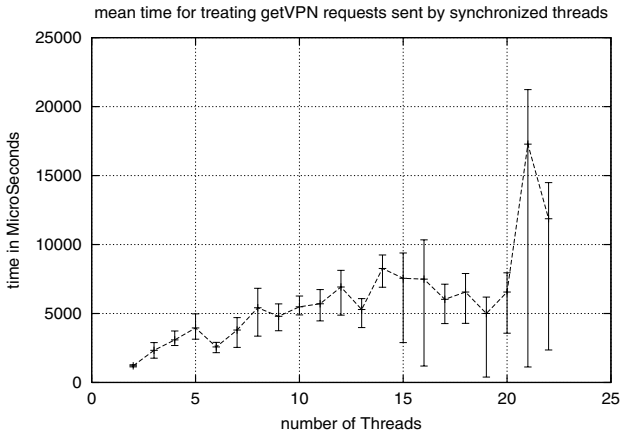

Figure 7. Processing time of GetVPN messages sent simultaneously by several clients.

We finally performed scalability tests. Figure 7 shows the time required to process a *GetVPN* SOAP message when the number of clients accessing the management interface increases. The results are good up to around 20 simultaneous clients, and then the average processing time largely increases. A first reason for this behavior is the fact that for each *GetVPN* message received, the management interface must invoke the database. Yet, since this is a shared database, the number of concurrent accesses is limited and some *GetVPN* messages cannot be handled immediately. A second reason that we suspect, is a negative impact of the Java and the J2EE platform on performances, but further investigation is needed here.

Our experiments have shown that the overall performance level is largely impacted by many different technological aspects. Yet we are confident that performance can still be largely improved, and efforts in this complex area are continuing. Therefore this section should be understood as a first evaluation of a prototype, not as a performance analysis of an optimized solution.

6. Related Works

Security remains an important barrier to customer adoption of web services, and in spite of many efforts, no single standard for security exists. Many documents and drafts have been written in this domain, yet the vast majority remains highly theoretical, as opposed to practical, real-world implementations.

The WS-Security document, which is now developed in the OASIS standards body, should be published for public comments and is expected to be completed within a few months. The WS-I (Web Services Interoperability) organization was formed recently and to promote open standards for web services interoperability and it will come up with security specifications, in particular WS-Security. Yet most standards require some time before they are truly useful from an interoperability point of view, and web services security standards are no exception.

As for management issues, no serious efforts have been done, and most of them are proprietary approaches from major application developers and providers. Lately a new OASIS Web Services Distributed Management (WSDM) Technical Committee was created as a way to bridge the gap between OASIS and other organizations such as the W3C Web Services Architecture Working Group and the Distributed Management Task Force (DMTF). This includes using web services architecture and technology to manage distributed resources. The first Web Services Distributed Management (WSDM) V1.0 Specification is expected in January 2004.

In addition to these organizations, business companies such as Microsoft, Actionl, Amberpoint, already realized the urgent need for web services management solutions. For instance Microsoft will soon release its new .NET-based management solution, Microsoft Operations Manager (MOM 2004).

Previous sections already discussed with great details several related works concerning security aspects of web services. This section does not re-open this discussion. A complementary question is: “why should we use web services while other middleware platforms like RMI, CORBA or DCOM provide great implementation vehicles for services”? The strengths of the web as an information distributor, namely simplicity of access and ubiquity, are important in resolving the fragmented middleware world where interoperability is hard to achieve. The web complements these platforms by providing a uniform and widely accessible interface over services that are more efficiently implemented in a traditional middleware platform.

7. Conclusions

Due to their flexibility and simplicity, web services gained much interest during the last few years. However the distributed and dynamic nature of web services requires advanced management capabilities. VPNs play a prominent role in the framework we propose to manage and secure these services, since VPNs provide the in-

frastructure over which business web services take place. The centralized Management Operation Point (MOP) performs the client authentication and authorization aspects, and manages policy and security configurations at the clients. Therefore it seamlessly takes in charge lots of hassles found in traditional web Services.

This work is supported by an implementation which shows the feasibility of the approach. Yet achieving a good performance level requires an appropriate tuning of many building blocks, which is a complex task. If we are not satisfied by some aspects, most problems are now identified and we are continuing our efforts.

References

[1] L. Alchaal, V. Roca, A. El-Sayed, and M. Habert. A vprn solution for fully secure and efficient group communications. July 2003. 8th IEEE Symposium on Computers and Communications (ISCC'03), Kemer - Antalya, Turkey.

[2] L. Alchaal, V. Roca, and M. Habert. Offering a multicast delivery service in a programmable secure ip vpn environment. Oct. 2002. Fourth International Workshop on Networked Group Communication (NGC'02), Boston, USA.

[3] Array Networks Inc. *SSL VPN vs IPSec VPN*, Jan. 2003. white paper.

[4] J. Barrett. *A Response to the Feature on IPv6 vs SSL*. Root Prompt Org., June 2000. http://rootprompt.org.

[5] Check Point Software Technologies Ltd. *IPSec Versus Clientless VPNs for Remote Access*, Sept. 2002. white paper, http://www.checkpoint.com.

[6] T. Dierks and C. Allen. *The TLS Protocol Version 1.0*, Jan. 1999. IETF Request for Comments, RFC 2246.

[7] S. Gokul. *XML Digital Signatures*. Inc. InformIT Division, Aug. 2002. http://www.informit.com.

[8] S. Gokul. *XML Encryption*. Inc. InformIT Division, Aug. 2002. http://www.informit.com.

[9] S. Gokul. *XML Key Management (XKMS)*. Inc. InformIT Division, Sept. 2002. http://www.informit.com.

[10] J. Harrison. *VPN Technologies - a comparison*. Data Connection Ltd., Feb. 2003. http://www.dataconnection.com.

[11] S. Kent and R. Atkinson. *Security Architecture for the Internet Protocol*, Nov. 1998. IETF Request for Comments, RFC 2401.

[12] Z. Kerraval. *The Evolution of Virtual Private Networks*, Oct. 2002. white paper, Yankee Group.

[13] D. Kosiur. *Building and Managing Virtual Private Networks*. John Wiley & Sons Inc., ISBN 0-471-29526-4, 1998.

[14] V. Machiraju, A. Sahai, and A. van Moorsel. Web service management network: An overlay network for federated service management. Aug. 2002. IEEE/IFIP IM 2003, Colorado Springs, USA.

[15] N. Mitra. *SOAP Version 1.2 Part 0: Primer*. W3C Working Group, Dec. 2002. work in progress.

[16] Netcelo S.A. http://www.netcelo.com.

[17] E. Newcomer. *Understanding Web Services: XML, WSDL, SOAP, and UDDI*. Addison Wesley Professional, ISBN 0-201-75081-3, 2002.

[18] OASIS Standard. *UDDI Spec Technical Committee Specification Version 3.0*, July 2002.

[19] OASIS Standard. *Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) v1.1*, Sept. 2003.

[20] OASIS Standard. *eXtensible Access Control Markup Language (XACML) version 1.0*, Feb. 2003.

[21] OASIS Standard. *Service Provisioning Markup Language (SPML) Version 1.0*, June 2003.

[22] A. Patrick. *SSL VPNs: The next hot mantra*. Network Computing company, Feb. 2003. http://www.nc-india.com/features/.

[23] W3C Standard. *Web Services Definition Language (WSDL)1.1*, Mar. 2001.