SOA Infrastructure Security Essentials

Securing the SOA Infrastructure of the Autonomous Province of Bolzano

Ver. 1.0

Official Documentation

SOA Infrastructure Security Essentials - Ver. 1.0

Securing the SOA Infrastructure of the Autonomous Province of Bolzano

14. Mar. 2013

Official Documentation

Dep. 9 - Department for Information Technology
Enterprise Architecture Group

Author:

Enterprise Architecture Group

Describes the essentials of the security infrastructure used in Web Service communication of the Service Oriented Architecture of the Autonomous Province of Bolzano

Content

1. Abstract

This document describes the technical aspects of the security infrastructure adopted for the **S**ervice **O**riented **A**rchitecture (SOA) of the Autonomous Province of Bolzano.

According to the official IT-security policy of the Autonomous Province of Bolzano, all participants of the SOA (both service consumers and servers offering the services) must support and implement what is described in the following chapters. The solution has been designed turning the most attention to integration of heterogeneous realities, reverting to official standards wherever possible.

2. Basic PROV.BZ-SOA Security Profile

The *Basic prov.bz-SOA Security Profile* addresses message authentication issues within the security context of the Province of Bolzano, trying to achieve the minimally requested security claims specified by the organization's security policy at lowest costs.

2.1. Inner-Realm Authentication

Our SOA environment requires any consumer of a service to prove to have been authenticated by a KDC of a unique Kerberos v.5 realm (which currently is Active Directory of Windows 2003 Server).

For securing Web Services, we use the Security extension of the SOAP header defined by the WSS Standard (*Web Services Security: SOAP Message Security 1.1*, standardized by Oasis). Authentication is proven by integrating a Kerberos Service Ticket into the header's Security extension according to the *Kerberos Token Profile 1.1* standard defined by Oasis[1]. If the account running the web site's processing extension can read it, the client has successfully shown to have authenticated.

It is up to the consumer to get the ticket for the destination server from the KDC[2] by providing an appropriate SPN (**S**ervice **P**rincipal **N**ame). In order to have working the entire SOA dynamically, we adopted the following convention for building SPNs dynamically:

`cee/%servicehost%:norealm.%realm%@%REALM%`

where:

- `%servicehost%:norealm` expands to the name of the web server offering the service without any specified port[3] and without the DNS domain. If the service's URL is, for example, http://myserver.mydomain.com:9701/myservice.asmx?wsdl, `%servicehost%:norealm` will expand to `myserver`.
- `%realm%` expands to the name of the consumer's Kerberos realm[4]. If your computer, for example, belongs to the Windows domain named *mydomain.com*, `%realm%` expands to `mydomain.com`.

The entire SPN string of our example would therefore be:
`cee/myserver.mydomain.com@MYDOMAIN.COM`.

---

[1] An example of a valid Security header element of a SOAP 1.2 can be found in Appendix A.

[2] Note that this ist true for all client processes, including a service which calls another one (whereby the calling service behaves as a client), see chapter *Service hopping* for further details.

[3] By omitting the TCP port, in our default scenario we cannot have running different services of our SOA with different service accounts on the same server.

[4] Our Kerberos realm is our Windows 2003 domain.

Note that the same SPN will result from the URL http://myserver:9701/myservice.asmx?wsdl. So, on the one hand the default SPN pattern makes service requests independent from whether the service's URL has been registered with the host's fully qualified domain name or with its plain server name only, on the other hand it prevents from requesting services from trusted Kerberos realms.

This approach allows for service authentication but not for identity impersonation in delegation scenarios which are described below.

2.1.1. Kerberos Delegation

If the services and service consumers are configured to support Kerberos delegation, they respect the following rules:

- they request the Kerberos service ticket to be forwardable
- they request the Kerberos service ticket for the HTTP service class with the following SPN: `http/%servicehost%:norealm:%port%@%REALM%` (in contrast to what described above),

where `%servicehost%:norealm` resolves to the name of the web server offering the service without the DNS domain name and `%port%` is omitted if it is either 80 or 443.

Running applications with support for Kerberos delegation is the recommended configuration since it does not compromise web servers that are not configured to support delegation as long as the HTTP service class for the destination server is registered with the same user that offers the service class as described in the scenario in the previous chapter.

2.1.2. Compatibility issues

An issue exists about compatibility between the Java and Active Directory:

- The default encryption algorithm for Kerberos tokens delivered by a Windows 2003 Domain Controller is the strongest supported, currently *RC4-HMAC* (128 bit) This encryption algorithm is not supported by the JAAS framework (in fact, the Kerberos protocol defines it as optional). So for service principals (users) running Java based services, the flag "*Use DES encryption types for this account*" has to be enabled in Active Directory in order to have tokens encrypted with the weaker algorithm (*DES-CBC-MD5* or *DES-CBC-CRC*, both of them 56 bit[5]).

2.2. Authenticating External Service Calls

While internal service calls prove to have authenticated by showing a valid Kerberos token, external service calls are authenticated by username and password. The credentials are integrated by the sender into the SOAP header according to the WSS Standard (*Web Services Security: SOAP Message Security 1.1*, standardized by Oasis) and the *UserName Token Profile 1.1*.

[5] *DES3-CBC-SHA1* is not supported by Windows 2003.

3. Appendix A: Example of a WSS Compliant Security Header Element for SOAP 1.2

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap=http://www.w3.org/2003/05/soap-envelope
   mlns:xsi=http://www.w3.org/2001/XMLSchema-instance mlns:xsd=http://www.w3.org/2001/XMLSchema
   mlns:wsa=http://schemas.xmlsoap.org/ws/2004/08/addressing mlns:wsse=http://docs.oasis-
   open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd

      <soap:Header>
           …
           <wsse:Security>
                <wsu:Timestamp
                     <wsu:Id="Timestamp-06bc75d0-3270-4790-ac0a-fdda04c78180">
                     <wsu:Created>2008-07-15T08:54:42Z</wsu:Created>
                     <wsu:Expires>2008-07-15T08:59:42Z</wsu:Expires>
                </wsu:Timestamp>
                <wsse:BinarySecurityToken ValueType=http://docs.oasis-open.org/wss/oasis-wss-
                   kerberos-token-profile-1.1#GSS_Kerberosv5_AP_REQ EncodingType=http://docs.oasis-
                   open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary
                   xmlns:wsu=http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
                   utility-1.0.xsd wsu:Id="SecurityToken-661fbb90-7b7c-4d3a-84f1-76a7477fa0c">
                …
                </wsse:BinarySecurityToken>
           </wsse:Security>

      </soap:Header>

      <soap:Body>
      …
      </soap:Body>

</soap:Envelope>
```

4. Reference

The following are references to the standards used for securing web services within the SOA of the Autonomous Province of Bolzano. You will have to adhere to these standards when implementing your own solution to access the SOA's services. You are expected to be familiar with web services.

Web Services Security: SOAP Message Security 1.1

http://www.oasis-open.org/committees/download.php/21257/wss-v1.1-spec-errata-os-SOAPMessageSecurity.htm

Kerberos Token Profile 1.1

http://www.oasis-open.org/committees/download.php/21250/wss-v1.1-spec-errata-os-KerberosTokenProfile.htm

UserName Token Profile 1.1

http://www.oasis-open.org/committees/download.php/16782/wss-v1.1-spec-os-UsernameTokenProfile.pdf