marcusevans conferences

Urbanisation des SI - Conduite du changement IT

20/03/09

Sécuriser ses Web Services

Patrick CHAMBET

http://www.chambet.com

Bouygues Telecom

Direction Gouvernance, Outils et Architecture / Sécurité du SI

Sommaire

- Vulnérabilités des Web Services
- Sécurisation des Web Services
  - Authentification des accès
  - Confidentialité et authenticité des messages
  - Audit et traçabilité des actions
- Recommandations d'implémentation

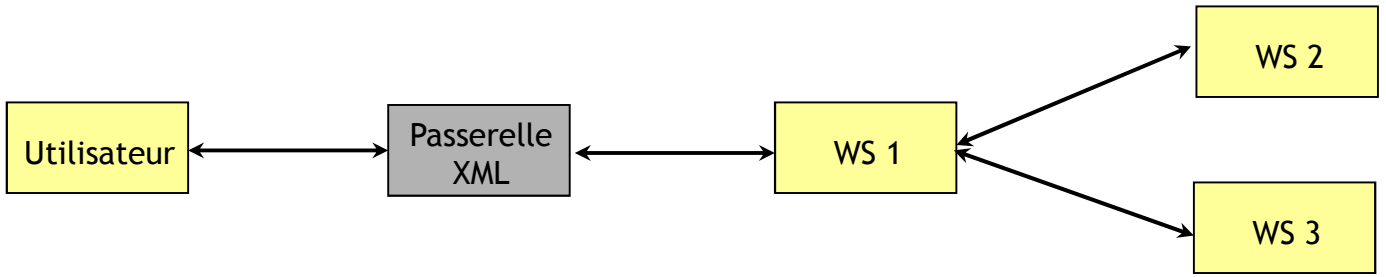
Rappel: les Web Services

- Une infrastructure d'échange applicatifs
- A base de messages XML
- Sur les protocoles « Web » (HTTP, HTTPS) en général
- Entre un client et des applications
  - A l'aide d'un simple browser Web
- Ou entre applications
- Interface décrite en langage WSDL (Web Services Description Language)
- Possibilité d'utiliser SOAP (Simple Object Access Protocol) pour l'échange des messages

- Très à la mode (cf SOA, SaaS, Cloud computing, …)

Web Services et sécurité (1/2)

- Avantages des Web Services
  - L'interopérabilité est prioritaire
  - L'hétérogénéité des applications disparaît: accès transparent à tout type de plate-forme
  - Briques indépendantes (les services)
  - Peuvent remplacer les bus (de type Tuxedo)



Web Services et sécurité (2/2)

- Un défi pour la sécurité
  - Traversent les firewalls et arrivent au cœur du SI
  - Sont souvent ouverts aux partenaires externes (ex: Rosettanet, …)
  - Services sans état par défaut (donc pas de sessions)
  - Pas de centralisation
  - Forte résistance des MOE face à la sécurisation de ce type d'architectures
- Comment authentifier les accès ?
- Comment protéger les données sensibles échangées ?
- Comment assurer la traçabilité des actions ?

Vulnérabilités des Web Services (1/2)

- Héritent des vulnérabilités des applications Web traditionnelles
  - Saisie de données hostiles
    - ➔ Buffer overflows, injections SQL, Cross Site Scripting (XSS), corruption de BD
  - Contournement de l'authentification, vol de session, tentatives de rejeu
  - Vulnérabilités des navigateurs
  - Cross Site Request Forgeries (CSRF)
- Possèdent leurs propres vulnérabilités
  - Envoi de code XML malformé ou XML overflow -> déni de service XML (XDoS)
    - ➔ Il existe des fuzzers pour XML (ex: http://untidy.sourceforge.net)
  - Inclusion dans le XML d'attachements MIME ou de binaires
    - ➔ Injections XPath
  - Injection de XML dans les champs XML -> changement de la structure et de la signification du message XML
  - Exécution de commandes sur le système d'exploitation

Vulnérabilités des Web Services (2/2)

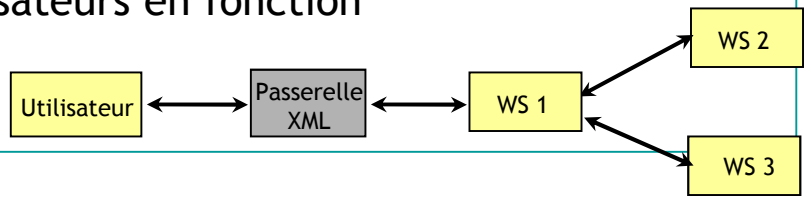
- Installations par défaut des progiciels
  - Mots de passe par défaut
  - Mots de passe en clair dans les fichiers de configuration
  - Login/mot de passe générique unique pour l'accès aux WS…
  - Y compris lorsque les WS sont exposés sur Internet !

Sommaire

- Vulnérabilités des Web Services
- Sécurisation des Web Services
  - Authentification des accès
  - Confidentialité et authenticité des messages
  - Audit et traçabilité des actions
- Recommandations d'implémentation

Sécurisation des accès aux WS

- Authentification des utilisateurs (ou des autres applications)
  - Login / mot de passe (le moins sécurisé)
    - ➔ Niveau transport ou message (transmission à d'autres WS)
  - Kerberos (en interne)
  - Jeton de session de type HTTP (mais on n'est plus sans état)
  - Certificats X.509
  - SAML (Security Assertion Markup Language): fédération d'identités avec des partenaires externes)
  - WS-Security
  - Lorsque rien de tout cela n'est possible: au minimum, filtrage sur les adresses IP sources des serveurs clients
- Contrôle d'accès aux fonctionnalités du Web Service
  - Gestion des permissions d'accès des utilisateurs en fonction de leur profil



SAML

- Standard de propagation de bout en bout du contexte de sécurité
- Basé sur XML
- Permet la description et l'échange des informations de sécurité (codes d'accès, privilèges, méthode d'authentification, ...) exprimées sous la forme d'assertions
- L'enregistrement, appelé jeton, est encapsulé dans SOAP et permet l'accès à des services sans nouvelle identification (SSO)
- Il définit pour chaque application et chaque individu, un objet (l'entité), et lui a associe des privilèges (les attributs)
- SAML est géré par l'OASIS

SAML: exemple de jeton

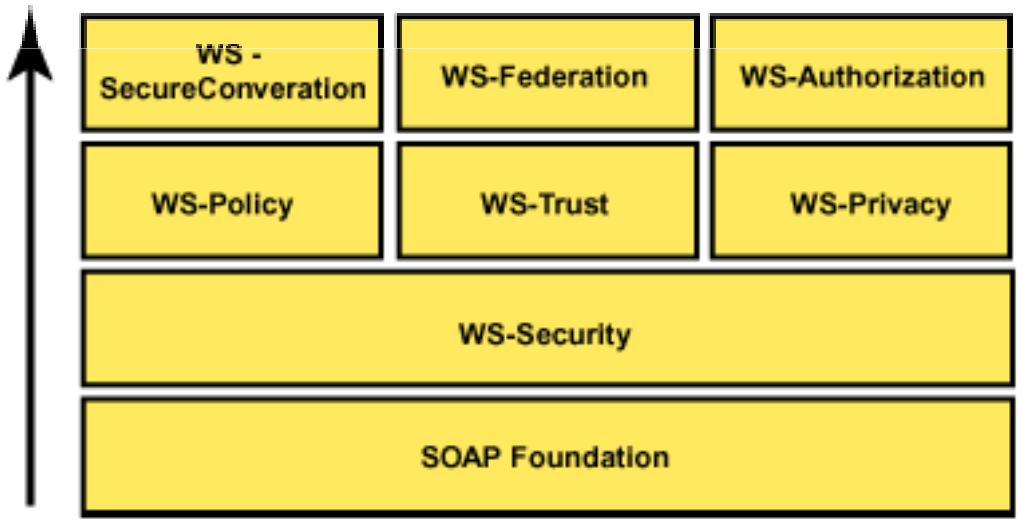
```
<saml:Conditions NotBefore="2009-03-03T18:12:33Z" NotOnOrAfter="2009-03-03T23:22:33Z">
  <saml:AudienceRestrictionCondition>
          <saml:Audience>https://www.mysite.com/</saml:Audience>
  </saml:AudienceRestrictionCondition>
</saml:Conditions>
<saml:AuthenticationStatement  AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:password"
      AuthenticationInstant="2009-03-03T20:29:23Z">
  <saml:Subject>
          <saml:NameIdentifier Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
                    jdupond@bbox.fr
          </saml:NameIdentifier>
  </saml:Subject>
</saml:AuthenticationStatement>
<saml:AttributeStatement>
  <saml:Subject> ... </saml:Subject>
  <saml:Attribute AttributeName="Group" AttributeNamespace="http://schemas.xmlsoap.org/claims">
          <saml:AttributeValue>
                    Utilisateur
          </saml:AttributeValue>
  </saml:Attribute>
</saml:AttributeStatement>
<Signature xmlns="http://www.w3.org/2000/09/xmldsig#"> ... </Signature>
```

WS-Security (1/2)

- WS-Security
  - Définit une série d'extensions SOAP
  - Plus large que SAML, intègre le traitement de l'authentification, du chiffrement et de l'intégrité des données
  - Accueille les spécifications tierces comme SAML, Kerberos, X.509, ...
  - Deux spécifications majeures ont été définies
    - ➔XML-Encryption : chiffrement total ou partiel des champs XML
    - ➔XML-Signature : authentification des interlocuteurs et intégrité du message

WS-Security (2/2)

- Autres spécifications de WS-Security



WS-Security: ex. de UserNameToken

```
<S11:Envelope xmlns:S11="..." xmlns:wsse="...">
  <S11:Header>
          <wsse:Security>
                    <wsse:UsernameToken>
                              <wsse:Username>jdupond</wsse:Username>
                              <wsse:Password>plouf</wsse:Password>
                    </wsse:UsernameToken>
          </wsse:Security>
  </S11:Header>
</S11:Envelope>
```

```
<S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu= "...">
  <S11:Header>
          <wsse:Security>
                    <wsse:UsernameToken>
                              <wsse:Username>jdupond</wsse:Username>
                              <wsse:Password Type="...#PasswordDigest">
                                        wfYI4nXd8PjNOVjsCJEV7t4rgHe4Rx==</wsse:Password>
                              <wsu:Created>2009-03-03T01:24:32Z</wsu:Created>
                    </wsse:UsernameToken>
          </wsse:Security>
  </S11:Header>
</S11:Envelope>
```

WS-Security: SOAP Message Security

```
<?xml version="1.0" encoding="utf-8"?>
<S11:Envelope xmlns:S11="..." xmlns:wsse="..." xmlns:wsu="..." xmlns:ds="...">
   <S11:Header>
       <wsse:Security>
            <wsse:BinarySecurityToken ValueType="...#X509v3" EncodingType="...#Base64Binary"
            wsu:Id="X509Token">MIIEZyCBA9CgBwIBAhIQEmtJXc0rqrJh5i...</wsse:BinarySecurityToken>
                <ds:Signature>
                        <ds:SignedInfo>
                            <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
                            <ds:Reference URI="#myBody">
                                <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
                                <ds:DigestValue>EZLdeytSo2...</ds:DigestValue>
                            </ds:Reference>
                        </ds:SignedInfo>
                        <ds:SignatureValue>BL7jdfTfEb2l/vXbMZPNjPOV...</ds:SignatureValue>
                        <ds:KeyInfo>
                            <wsse:SecurityTokenReference>
                                <wsse:Reference URI="#X509Token"/>
                            </wsse:SecurityTokenReference>
                        </ds:KeyInfo>
            </ds:Signature>
       </wsse:Security>
   </S11:Header>
   <S11:Body wsu:Id="myBody">
   ...
   </S11:Body>
</S11:Envelope>
```

Ex. d'implémentation (Weblogic)

```
package examples.webservices.security_jws.client;

import weblogic.security.SSL.TrustManager;
import weblogic.xml.crypto.wss.provider.CredentialProvider;
import weblogic.xml.crypto.wss.WSSecurityContext;
import weblogic.wsee.security.bst.ClientBSTCredentialProvider;
import weblogic.wsee.security.unt.ClientUNTCredentialProvider;
(...)
import java.security.cert.X509Certificate;

public class SecureHelloWorldClient {
  public static void main(String[] args) throws Throwable {

    //username or password for the UsernameToken
    String username = args[0];
    String password = args[1];
    //client private key file
    String keyFile = args[2];
    //client certificate
    String clientCertFile = args[3];
    String wsdl = args[4];

    SecureHelloWorldService service = new SecureHelloWorldService_Impl(wsdl + "?WSDL" );
    SecureHelloWorldPortType port = service.getSecureHelloWorldServicePort();

    //create credential provider and set it to the Stub
    List credProviders = new ArrayList();

    //client side BinarySecurityToken credential provider -- x509
    CredentialProvider cp = new ClientBSTCredentialProvider(clientCertFile, keyFile);
    credProviders.add(cp);

    //client side UsernameToken credential provider
    cp = new ClientUNTCredentialProvider(username, password);
    credProviders.add(cp);

    Stub stub = (Stub)port;
    stub._setProperty(WSSecurityContext.CREDENTIAL_PROVIDER_LIST, credProviders);
    (...)
  }
}
```

Protection des données échangées

- Chiffrement du flux
  - HTTPS (SSL/TLS), IPSec
- Chiffrement des champs XML
  - Assure la confidentialité des données
- Signature des champs XML
  - Assure l'intégrité des données (non répudiation également)

- Attention: le chiffrement ne protège pas contre les intrusions !

Durcissement applicatif

- Changer les mots de passe par défaut
- Ne pas faire confiance aux données XML reçues
  - Filtrage des données côté serveur
  - Validation du schéma XML et du contenu des champs
  - Analyse du contenu
- Des relais filtrants (« firewalls XML » ou « firewalls applicatifs ») sont spécialisés dans la validation de données XML
  - Ex: IBM DataPower, DenyAll, BeeWare, …
  - Sous forme d'appliances ou de services à installer sur un serveur
  - Assurent aussi l'authentification et le chiffrement SSL

Traçabilité des actions (1/2)

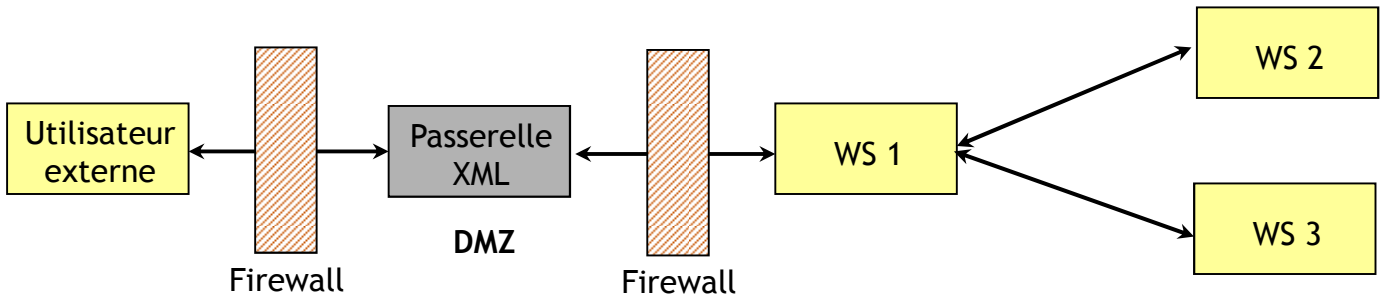
- Les actions effectuées doivent être tracées
  - C'est parfois une obligation légale (ex: accès Internet pour un FAI)
  - Sinon, cela permet de se couvrir en cas d'enquête
- Gestion de journaux d'événements / de logs
- Enregistrement pour chaque appel à un Web Service
  - Utilisateur
  - Action effectuée
  - Données sensibles manipulées

Traçabilité des actions (2/2)

- Si pas de possibilité de logs applicatifs
  - Logs du serveur HTTP (requêtes GET / POST, utilisateur, adresse IP source, …)
  - Logs des firewalls
  - Logs des proxies HTTP
  - Logs des firewalls applicatifs XML

Recommandations d'implémentation (1/2)

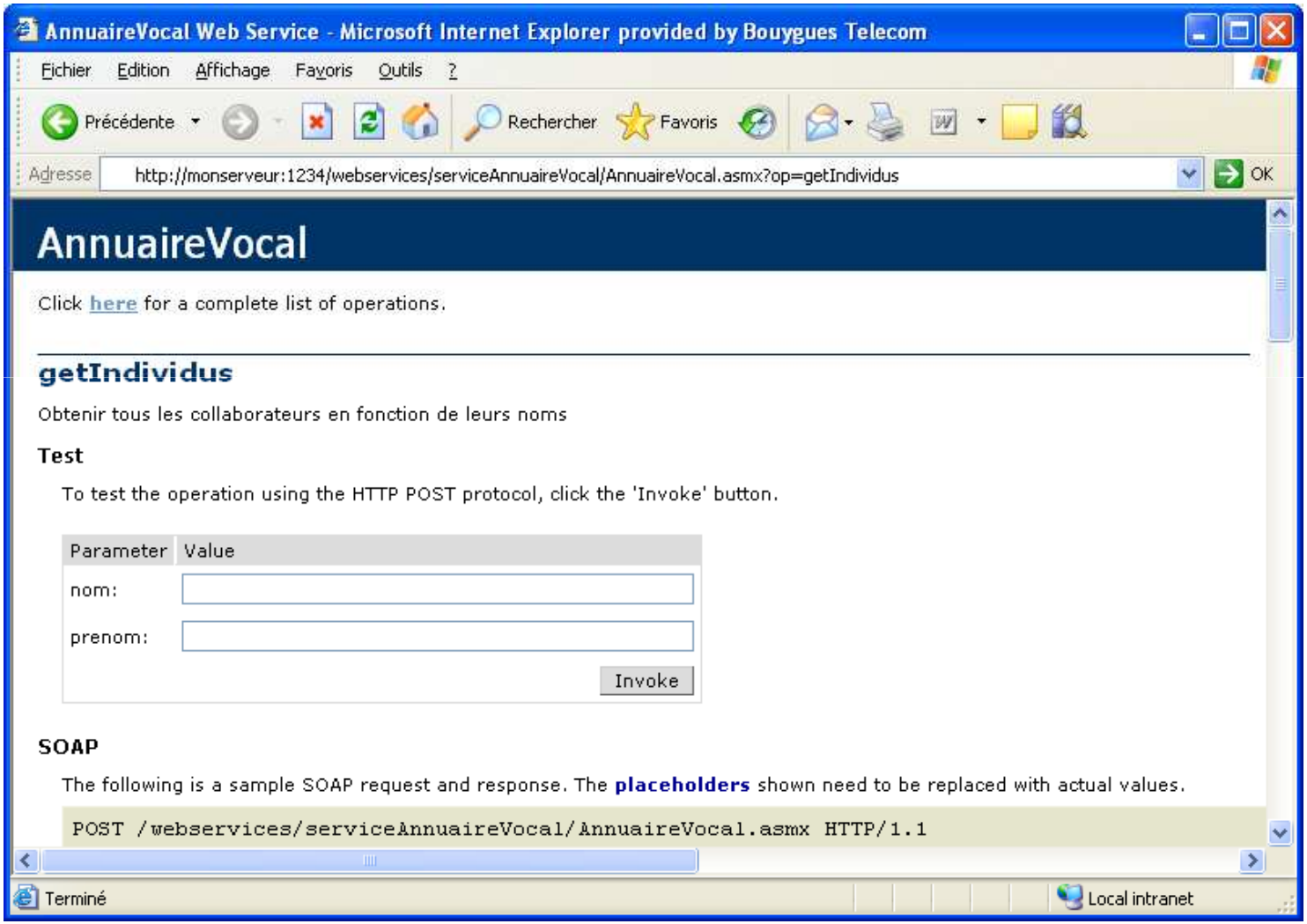
- Intégrer les bonnes pratiques dans les normes de sécurité des développements informatiques de l'entreprise
- Faire particulièrement attention aux accès externes qui arrivent au cœur du SI
  - Utiliser une architecture d'accès sécurisée
  - Passer par des relais de filtrage applicatif situés en DMZ
  - Filtrer sur les adresses sources des partenaires externes quand c'est possible (au niveau des firewalls externes)



Recommandations d'implémentation (2/2)

- Utiliser les fonctionnalités d'authentification et de filtrage des données proposées en standard par les environnements de développement et les serveurs d'applications
  - Ex: Weblogic supporte l'intégration Kerberos Windows
- Ne pas publier son catalogue WSDL
  - Surtout à l'extérieur (sur Internet)
    - ➔ Ex: chercher "filetype:wsdl" sur Google…
  - Y compris dans les annuaires UDDI (Universal Description Discovery and Integration)
- Séparer le réseau d'administration des Web Services du réseau de production

Exemple de description WSDL



Conclusion

- La sécurisation des Web Services est un sujet complexe
- Mais du fait du succès des Web Services, leur sécurisation est de plus en plus incontournable dans la plupart des SI
- La protection des données échangées est souvent une obligation légale
- C'est indissociable de la prise en compte de la sécurité dans les projets informatiques
- Un grand nombre d'acteurs sont impliqués, de bout en bout des processus de l'entreprise

Pour aller plus loin…

- Vulnérabilités des applications Web
  - http://www.chambet.com/publications/sec-web-apps/
- OASIS
  - http://www.oasis-open.org/committees/wss/
  - http://www.oasis-open.org/news/oasis-news-2009-02-05.php
- Microsoft Web Services Security Patterns
  - http://msdn.microsoft.com/en-us/library/aa480545.aspx
- Securing Weblogic Web Services
  - http://download.oracle.com/docs/cd/E12840_01/wls/docs103/webserv_sec/
- SAP
  - https://www.sdn.sap.com/irj/scn/articles-webservices-all
- OpenLiberty secure identity Web Services
  - http://www.openliberty.org/

Questions