Integración Sistema CEVALDOM

Seguridad en los Web Service

Versión: 1.2

WEB SERVICE

Un Web Service (Servicio Web) es una aplicación que utiliza un conjunto de protocolos y estándares que sirven para intercambiar información entre aplicaciones, pero éstas informaciones necesitan transportarse de manera segura, ya que cualquier persona con intención maliciosa podría interceptar las peticiones a los servicios. Para evitar esto, **CEVALDOM** implementará la siguiente solución:

Se utilizará el método **HMAC/SHA-256** (Hash Message Authentication Code) que consiste en un secreto compartido entre **CEVALDOM** y el **CLIENTE** para poder realizar las peticiones al servicio de manera que esta no pueda ser interceptada y reproducida para realizar ataques por terceros.

IMPLEMENTACION

1. **CEVALDOM** asignará un **USUARIO** y Diez (10) **CODE** únicos de Cuatro (4) dígitos a cada cliente. El cliente tendrá la obligación de utilizar uno de estos códigos en cada petición utilizándolo como llave (secretKey) para la encriptación. Para la creación del mensaje encriptado, el cliente tendrá que concatenar su **USUARIO** de acceso a los servicios más la fecha actual (**DD/MM/YYY HH:MM:SS**) para así generar el **HMAC/SHA-256** y enviarlo como dato en la petición.

   Luego de generar el mensaje encriptado (**HMAC/SHA-256**) el cliente debe de enviar los siguientes parámetros:

   a. **CODE:** Se refiere al número (**0 al 9**) del código secreto utilizado para la generación del **HMAC/SHA-256**. Ejemplo **CODE**: 3, donde el código número asignado por **CEVALDOM** y utilizado para generar el **HMAC/SHA-256** es **5030**.
   b. **USER:** Se refiere al usuario de acceso proporcionado por **CEVALDOM** al cliente.
   c. **DATE:** Es la fecha que fue generado el **HMA/SHA-C256.**
   d. **TOKEN:** Se refiere al mensaje encriptado que fue generado por el **HMAC/SHA-256.**

2. **CEVALDOM** al momento de recibir una petición a algún servicio, validará lo siguiente:

   a. Que el **USUARIO** que está haciendo la petición tenga acceso al servicio solicitado.
   b. Que el **CODE** suministrado no haya sido utilizado en la petición anterior.
   c. **CEVALDOM** recrea el **TOKEN** que generó el cliente y comprueba que el **TOKEN** generado es legítimo.
   d. Ya con estas validaciones correctas **CEVALDOM** considerará válida la petición y le concederá al solicitante acceso a la información.

EJEMPLO

Suponiendo que entre los 10 códigos proporcionados por **CEVALDOM** el código número **3** es **5030**, para generar el **TOKEN** tenemos lo siguiente:

- **codeNumber** = 5030
- **userLogin** = CVDMADM
- **generationDate** = 27/04/2016 05:04:44

El mensaje es la concatenación de las variables **userLogin** + **generationDate** y el secretKey es el **codeNumber**, ejemplo:

- **Message** = CVDMADM27/04/2016 05:04:44
- **secretKey** = 5030
- **generatedToken** = **4d462d7732d61afa6a7ee700dc60ba496ec36a2b**

A continuación un ejemplo de los datos que contendría una petición realizada a un servicio de **CEVALDOM:**

- **USER =** CVDMADM
- **CODE** = 3
- **DATE** = 27/04/2016 05:04:44
- **TOKEN** = 4d462d7732d61afa7ee700dc60ba496ec36a2b

IMPLEMENTACION EN JAVA

```
// Obtener los bytes de la clave proporcionada por CEVLADOM
byte[] keyBytes = codeNumber.getBytes();
SecretKeySpec signingKey = new SecretKeySpec(keyBytes, "HmacSHA256");

// Obtenemos una instancia Mac hmac_sha256 e inicializar con la clave de firma
Mac mac = Mac.getInstance("HmacSHA256");
mac.init(signingKey);

String message = userLogin + generationDate;
// Calculamos el MAC de bytes de la variable message
byte[] rawHmac = mac.doFinal((message).getBytes());

// Convertimos raw bytes to Hex
byte[] hexBytes = new Hex().encode(rawHmac);

// Convertimos el arreglo de bytes a String
String token = new String(hexBytes, "UTF-8");
```