Guía de migración a firma HMAC SHA256
Conexión por Redirección

Versión: 1.0

06/10/2015

Redsys · C/ Francisco Sancha, 12 · 28034 · Madrid · ESPAÑA

Autorizaciones y control de versión

Versión	Fecha	Afecta	Breve descripción del cambio
1.0	06/10/2015		Versión inicial del documento

ÍNDICE DE CONTENIDO

1. Introducción

1.1 Objetivo

Este documento recoge los aspectos técnicos necesarios para que un comercio, que actualmente esté operando en el SIS, realice la migración con el TPV Virtual utilizando el nuevo sistema de firma basado en HMAC SHA256.

El algoritmo actual (SHA-1) en el que se basa la seguridad de la conexión de la tienda con el tpv virtual (y viceversa) es un algoritmo obsoleto según los estándares de seguridad, y podría ser objeto de ataques. Para garantizar la mayor seguridad en la conexión con el TPV Virtual SIS los métodos anteriores deben actualizarse al nuevo sistema de firma basado en HMAC SHA256.

Esta documentación aplica a los comercios que acceden al SIS por redirección del navegador del cliente/comprador, (entrada "Realizar Pago") y también afecta a las notificaciones desde el TPV Virtual hacia el servidor del comercio para comunicación de los resultados (modalidades "Síncrona" o "Asíncrona")

Para desarrollar el cálculo de este nuevo tipo de firma, el comercio puede realizar el desarrollo por sí mismo utilizando las funciones estándar o utilizar las librerías suministradas (PHP y JAVA) cuya utilización se presenta en detalle en esta guía y que están a su disposición en:

http://www.redsys.es/wps/portal/redsys/publica/areadeserviciosweb/descargaDeDocumentacionYEjecutables/

1.2 Definiciones, siglas y abreviaturas

SIS. Servidor Integrado de Redsys.

1.3 Referencias

- Documentación de Integración con el SIS
- Guía de comercios del SIS.

2. Resumen de las diferencias del nuevo modelo (HMAC SHA-256) respecto del modelo anterior (SHA-1)

Las diferencias del nuevo modelo de conexión basado en HMAC SHA-256 respecto al modelo anterior en uso (basado en SHA-1) se pueden resumir en los siguientes 3 puntos:

A. **Conexión desde la tienda web hacia el TPV Virtual** para iniciar de una operación:
   1. Formato de los parámetros enviados.
      - **ANTES**: En el modelo anterior los parámetros eran enviados como campos independientes de un formulario.
      - **AHORA**: Los parámetros se agrupan en formato JSON y se envían como un único campo en el formulario.
   2. Cálculo de la firma enviada por la tienda web.
      - **ANTES**: La firma se calculaba como un SHA-1 sobre la concatenación de los campos enviados. No se enviaba ningún parámetro indicador de la versión de firma utilizada.
      - **AHORA**: La firma se calcula con una nueva clave diversifica por operación, y un nuevo algoritmo (HMAC SHA256). Se firma la lista de parámetros enviados en formato JSON. Además se incluye un campo nuevo que indica la versión de firma utilizada.

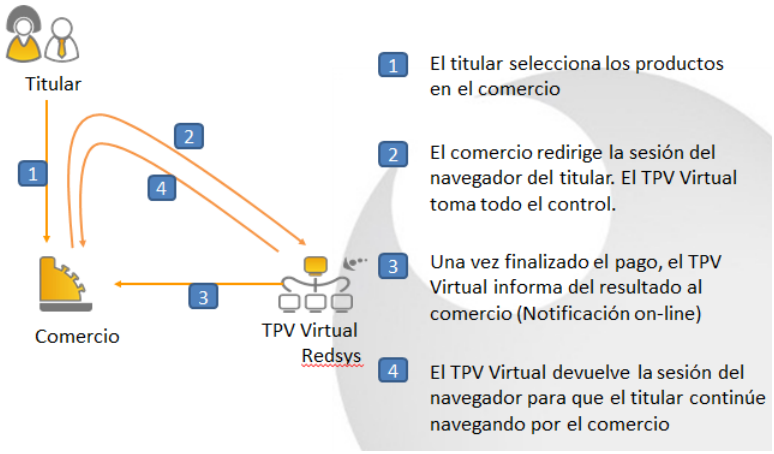
B. **Retorno de la navegación del cliente desde el TPV virtual a la web de la tienda**. Los cambios son los mismos descritos en el punto A).

C. **Envío de notificaciones on-line desde el TPV virtual** al servidor de la tienda web.
   1. Comercios con notificaciones HTTP (tanto asíncrona como síncrona): Los cambios son los mismos descritos en el punto A).
   2. Comercios con notificaciones SOAP:
      - **ANTES**: La cadena de campos del mensaje enviado se firma con algoritmo basado en SHA-1.
      - **AHORA**: La cadena de campos del mensaje enviado se firma con la nueva clave y utilizando un nuevo algoritmo (HMAC SHA256).

NOTA: No es necesario modificar ningún parámetro dentro del módulo de administración del comercio. El TPV Virtual aceptará de forma automática las conexiones basadas en HMAC SHA-256, una vez que la tienda online empiece a enviarlas al TPV Virtual. De igual forma, el TPV Virtual utilizará el formato HMAC SHA-256 para confirmar dichas operaciones al servidor de la tienda (notificaciones y retorno de la navegación del cliente).

3. Descripción general del flujo

El siguiente esquema presenta el flujo general de una operación realizada con el TPV Virtual.



3.1 Envío de petición al TPV Virtual

Como se muestra en el paso 2 del esquema anterior, el comercio debe enviar al TPV Virtual los datos de la petición de pago a través del navegador del titular. Para ello deberá preparar un formulario con los siguientes campos:

- **Ds_SignatureVersion**: Constante que indica la versión de firma que se está utilizando.
- **Ds_MerchantParameters**: Cadena en formato JSON con todos los parámetros de la petición codificada en Base 64 y sin retornos de carro (En el Anexo 1 del apartado Anexos del presente documento se incluye la lista de parámetros que se pueden enviar en una solicitud de pago). Entre los parámetros que se pueden enviar ya no tiene cabida el parámetro "Ds_Merchant_MerchantSignature", como se puede comprobar en el apartado Anexos.
- **Ds_Signature**: Firma de los datos enviados. Resultado del HMAC SHA256 de la cadena JSON codificada en Base 64 enviada en el parámetro anterior. Este nuevo parámetro es la firma del comercio en sustitución del parámetro "Ds_Merchant_MerchantSignature" que se enviaba anteriormente.

Este formulario debe enviarse a las siguientes URLs dependiendo de si se quiere realizar una petición de pruebas u operaciones reales:

URL Conexión	Entorno
https://sis-t.redsys.es:25443/sis/realizarPago	Pruebas
https://sis.redsys.es/sis/realizarPago	Real

3.2 Recepción del resultado (Notificación on-line)

Una vez gestionado el pago, el TPV Virtual puede informar al servidor del comercio el resultado de la misma mediante una conexión directa al servidor del comercio (paso 3 del flujo descrito). Esta notificación es opcional y para su recepción debe estar configurada para cada terminal en el Modulo de Administración.

La notificación on-line consiste en un POST HTTP con la información del resultado. En el POST se incluirán los siguientes campos:

- **Ds_SignatureVersion**: Constante que indica la versión de firma que se está utilizando.
- **Ds_MerchantParameters:** Cadena en formato JSON con todos los parámetros de la respuesta codificada en Base 64 y sin retornos de carro (En el Anexo 2 del apartado Anexos del presente documento se incluye la lista de parámetros que se pueden incluir en la notificación on-line). Entre los parámetros que forman parte de este campo ya no tiene cabida el parámetro "Ds_Signature" que se utilizaba anteriormente como firma del comercio, como se puede observar en Anexos.
- **Ds_Signature**: Firma de los datos enviados. Resultado del HMAC SHA256 de la cadena JSON codificada en Base 64 enviada en el parámetro anterior. **El comercio es responsable de validar el HMAC enviado por el TPV Virtual para asegurarse de la validez de la respuesta. Esta validación es necesaria para garantizar que los datos no han sido manipulados y que el origen es realmente el TPV Virtual.**

NOTA: El TPV Virtual envia la notificación on-line a la URL informada por el comercio en el parámetro Ds_Merchant_MerchantURL.

3.3 Retorno del control de la navegación del titular

En el paso 4 del flujo el TPV Virtual devuelve al comercio el control de la navegación del cliente. De esta forma el comercio puede completar el flujo del pago manteniendo una secuencia de navegación natural para el cliente/comprador.

Opcionalmente el TPV Virtual puede incluir los mismos campos de la notificación on-line.

4. Formulario de envío de petición

El comercio deberá montar un formulario con los parámetros de la petición de pago que debe hacer llegar al TPV Virtual a través del navegador del cliente.

En el antiguo acceso (basado en SHA-1) los parámetros se enviaban de forma individual, tal y como se muestra en el siguiente ejemplo:

```
<form action="https://sis.redsys.es/sis/realizarPago" method="POST" >
        <input type="text" name="Ds_Merchant_Amount"
        value="145"/>
        <input type="text" name="Ds_Merchant_Order"
        value="1442772645"/>
        …
        …
        …
        <input type="submit" value="Realizar Pago"/>
</form>
```

A continuación se muestra un ejemplo del formulario de la nueva petición de pago (se puede observar que los campos se envían agrupados en formato JSON y empaquetados en Base64 sin retornos de carro):

```
<form action="https://sis.redsys.es/sis/realizarPago" method="POST" >
        <input type="text" name="Ds_SignatureVersion"
        value="HMAC_SHA256_V1"/>
        <input type="text" name="Ds_MerchantParameters"
        value="eyJEU19NRVJDSEFOVF9BTU9VTlQiOiIxNDUiLCJEU19NRVJDSEFOVF9PUkRFUiI6Ij
        E0NDI3NzI2NDUiLCJEU19NRVJDSEFOVF9NRVJDSEFOVENPREUiOiI5OTkwMDg4ODEiLCJ
        EU19NRVJDSEFOVF9DVVJSRU5DWSI6Ijk3OCIsIkRTX01FUkNIQU5UX1RSQU5TQUNUSU
        9OVFlQRSI6IjAiLCJEU19NRVJDSEFOVF9URVJNSU5BTCI6Ijg3MSIsIkRTX01FUkNIQU5UX
        01FUkNIQU5UVVJMIjoiaHR0cHM6XC9cL2VqZW1wbG9cL2VqZW1wbG9fVVJMX05vdGlmL
        nBocCIsIkRTX01FUkNIQU5UX1VSTE9LIjoiaHR0cHM6XC9cL2VqZW1wbG9cL2VqZW1wbG
        9fVVJMX09LX0tPLnBocCIsIkRTX01FUkNIQU5UX1VSTEtPIjoiaHR0cHM6XC9cL2VqZW1wb
        G9cL2VqZW1wbG9fVVJMX09LX0tPLnBocCJ9"/>
        <input type="text" name="Ds_Signature"
        value="hueCwD/cbvrCi+9IDY86WteMpXulIl0IDNXNlYgcZHM="/>
        <input type="submit" value="Realizar Pago"/>
</form>
```

Para facilitar la adaptación del comercio al nuevo tipo de firma, a continuación se explica de forma detallada los pasos a seguir para montar el formulario de petición de pago.

4.1 Identificar la versión de algoritmo de firma a utilizar

En el modelo de conexión antiguo (SHA-1) no se incluía ningún parámetro identificando la versión de firma. En el nuevo formato, en la petición se debe identificar la versión concreta de algoritmo que se está utilizando para la firma. Actualmente se utiliza el valor **HMAC_SHA256_V1** para identificar la versión de todas las peticiones, por lo que este será el valor del parámetro **Ds_SignatureVersion**, tal y como se puede observar en el ejemplo de formulario mostrado al inicio del apartado 3.

4.2 Montar la cadena de datos de la petición

Se debe montar una cadena con todos los datos de la antigua petición de pago en formato JSON. JSON es un formato abierto de intercambio de datos basado en texto. Al igual que el XML está diseñado para ser legible e independiente de la plataforma tecnológica. La codificación de datos en JSON es muy ligera por lo que es ideal para intercambio de datos en aplicaciones Web.

El nombre de cada parámetro debe indicarse en mayúsculas o con estructura "CamelCase" (Por ejemplo: DS_MERCHANT_AMOUNT o Ds_Merchant_Amount). La lista de parámetros que se pueden incluir en la petición se describe en el Anexo 1(Datos de la petición de pago) del apartado Anexos del presente documento. A continuación se muestra un ejemplo del objeto JSON de una petición:

{"DS_MERCHANT_AMOUNT":"145","DS_MERCHANT_ORDER":"1442772645","DS_MERCHANT_MERCHANTCODE":"999008881","DS_MERCHANT_CURRENCY":"978","DS_MERCHANT_TRANSACTIONTYPE":"0","DS_MERCHANT_TERMINAL":"871","DS_MERCHANT_MERCHANTURL":"https://ejemplo/ejemplo_URL_Notif.php","DS_MERCHANT_URLOK":"https://ejemplo/ejemplo_URL_OK_KO.php","DS_MERCHANT_URLKO":"https://ejemplo/ejemplo_URL_OK_KO.php"}

Como se puede observar, el conjunto de parámetros que forma el objeto JSON debe ser exactamente igual que el conjunto de parámetros que se enviaba de manera individual por el antiguo acceso, a excepción del parámetro Ds_Merchant_MerchantSignature que no forma parte del objeto JSON, tal y como se ha comentado con anterioridad.

Una vez montada la cadena JSON con todos los campos, es necesario codificarla en BASE64 sin retornos de carro para asegurarnos de que se mantiene constante y no es alterada en su paso por el navegador del titular.

A continuación se muestra el objeto JSON que se acaba de mostrar codificado en BASE64:

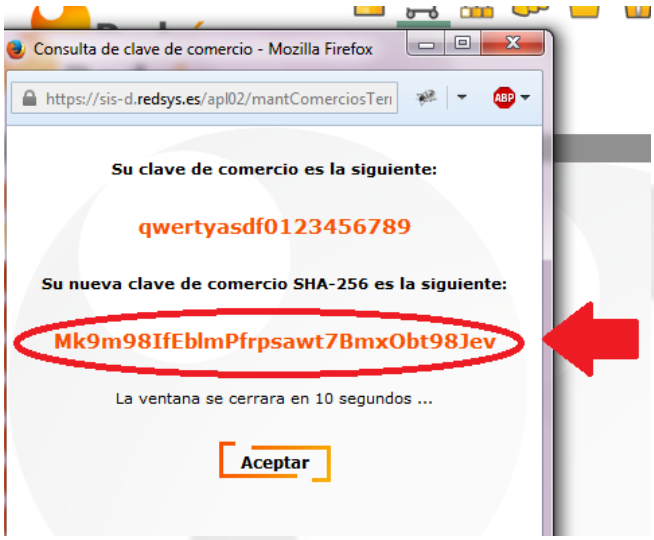
eyJEU19NRVJDSEFOVF9BTU9VTlQiOiIxNDUiLCJEU19NRVJDSEFOVF9PUkRFUiI6IjE0NDI3NzI2NDUiLCJEU19NRVJDSEFOVF9NRVJDSEFOVENPREUiOiI5OTkwMDg4ODEiLCJEU19NRVJDSEFOVF9DVVJSRU5DWSI6Ijk3OCIsIkRTX01FUkNIQU5UX1RSQU5TQUNUSU9OVFlQRSI6IjAiLCJEU19NRVJDSEFOVF9URVJNSU5BTCI6Ijg3MSIsIkRTX01FUkNIQU5UX01FUkNIQU5UVVJMIjoiaHR0cHM6XC9cL2VqZW1wbG9cL2VqZW1wbG9fVVJMX05vdGlmLnBocCIsIkRTX01FUkNIQU5UX1VSTE9LIjoiaHR0cHM6XC9cL2VqZW1wbG9cL2VqZW1wbG9fVVJMX09LX0tPLnBocCIsIkRTX01FUkNIQU5UX1VSTEtPIjoiaHR0cHM6XC9cL2VqZW1wbG9cL2VqZW1wbG9fVVJMX09LX0tPLnBocCJ9

La cadena resultante de la codificación en BASE64 será el valor del parámetro **Ds_MerchantParameters**, tal y como se puede observar en el ejemplo de formulario mostrado al inicio del apartado 3.

NOTA: La utilización de las librerías de ayuda proporcionadas por Redsys para la generación de este campo, se expone en el apartado 3.5.

4.3 Identificar la clave a utilizar para la firma

Para calcular la firma es necesario utilizar una clave específica para cada terminal. Se puede obtener la clave accediendo al Módulo de Administración, opción Consulta datos de Comercio, en el apartado "Ver clave", tal y como se muestra en la siguiente imagen:



NOTA IMPORTANTE: Esta clave debe ser almacenada en el servidor del comercio de la forma más segura posible para evitar un uso fraudulento de la misma. El comercio es responsable de la adecuada custodia y mantenimiento en secreto de dicha clave.

4.4 Firmar los datos de la petición

Una vez se tiene montada la cadena de datos a firmar y la clave específica del terminal se debe calcular la firma siguiendo los siguientes pasos:

1. Se genera una clave específica por operación. Para obtener la clave derivada a utilizar en una operación se debe realizar un cifrado 3DES entre la clave del comercio y el valor del número de pedido de la operación (Ds_Merchant_Order).

2. Se calcula el HMAC SHA256 del valor del parámetro **Ds_MerchantParameters** y la clave obtenida en el paso anterior.

3. El resultado obtenido se codifica en BASE 64, y el resultado de la codificación será el valor del parámetro **Ds_Signature**, tal y como se puede observar en el ejemplo de formulario mostrado al inicio del apartado 3.

NOTA: La utilización de las librerías de ayuda proporcionadas por Redsys para la generación de este campo, se expone en el apartado 3.5.

4.5 Utilización de librerías de ayuda

En los apartados anteriores se ha descrito la forma de acceso al SIS utilizando conexión por Redirección y el sistema de firma basado en HMAC SHA256. En este apartado se explica cómo se utilizan las librerías disponibles en PHP y JAVA para facilitar los desarrollos y la generación de los campos del formulario de pago. El uso de las librerías suministradas por Redsys es opcional, si bien simplifican los desarrollos a realizar por el comercio.

4.5.1 Librería PHP

A continuación se presentan los pasos que debe seguir un comercio para la utilización de la librería PHP proporcionada por Redsys:

1. Importar el fichero principal de la librería, tal y como se muestra a continuación:

```
include_once 'redsysHMAC256_API_PHP_4.0.2/apiRedsys.php';
```

   El comercio debe decidir si la importación desea hacerla con la función "include" o "required", según los desarrollos realizados.

2. Definir un objeto de la clase principal de la librería, tal y como se muestra a continuación:

```
$miObj = new RedsysAPI;
```

3. Calcular el parámetro **Ds_MerchantParameters**. Para llevar a cabo el cálculo de este parámetro, inicialmente se deben añadir todos los parámetros de la petición de pago que se desea enviar, tal y como se muestra a continuación:

```
$miObj->setParameter("DS_MERCHANT_AMOUNT",$amount);
$miObj->setParameter("DS_MERCHANT_ORDER",$id);
$miObj->setParameter("DS_MERCHANT_MERCHANTCODE",$fuc);
$miObj->setParameter("DS_MERCHANT_CURRENCY",$moneda);
$miObj->setParameter("DS_MERCHANT_TRANSACTIONTYPE",$trans);
$miObj->setParameter("DS_MERCHANT_TERMINAL",$terminal);
$miObj->setParameter("DS_MERCHANT_MERCHANTURL",$url);
$miObj->setParameter("DS_MERCHANT_URLOK",$urlOK);
$miObj->setParameter("DS_MERCHANT_URLKO",$urlKO);
```

Por último, para calcular el parámetro **Ds_MerchantParameters**, se debe llamar a la función de la librería "createMerchantParameters()", tal y como se muestra a continuación:

```
$params = $miObj->createMerchantParameters();
```

4. Calcular el parámetro **Ds_Signature**. Para llevar a cabo el cálculo de este parámetro, se debe llamar a la función de la librería "createMerchantSignature()" con la clave obtenida del módulo de administración, tal y como se muestra a continuación:

```
$claveModuloAdmin = 'Mk9m98IfEblmPfrpsawt7BmxObt98Jev';
$signature = $miObj->createMerchantSignature($claveModuloAdmin);
```

5. Una vez obtenidos los valores de los parámetros **Ds_MerchantParameters** y **Ds_Signature** , se debe rellenar el formulario de pago con dichos valores, tal y como se muestra a continuación:

```
<form action="https://sis.redsys.es/sis/realizarPago"
    method="POST" target="_blank">

    <input type="text" name="Ds_SignatureVersion"
        value="HMAC_SHA256_V1"/>
    <input type="text" name="Ds_MerchantParameters"
        value="<?php echo $params; ?>"/>
    <input type="text" name="Ds_Signature"
        value="<?php echo $signature; ?>"/>
    <input type="submit" value="Realizar Pago"/>

</form>
```

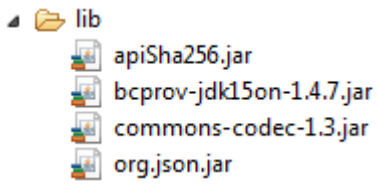
4.5.2 Librería JAVA

A continuación se presentan los pasos que debe seguir un comercio para la utilización de la librería JAVA proporcionada por Redsys:

1. Importar la librería, tal y como se muestra a continuación:

```
<%@page import="sis.redsys.api.ApiMacSha256"%>
```

El comercio debe incluir en la vía de construcción del proyecto todas las librerías(JARs) que se proporcionan:



2. Calcular el parámetro **Ds_MerchantParameters**. Para llevar a cabo el cálculo de este parámetro, inicialmente se deben añadir todos los parámetros de la petición de pago que se desea enviar, tal y como se muestra a continuación:

```
ApiMacSha256.setParameter("DS_MERCHANT_AMOUNT", amount);
ApiMacSha256.setParameter("DS_MERCHANT_ORDER", id);
ApiMacSha256.setParameter("DS_MERCHANT_MERCHANTCODE", fuc);
ApiMacSha256.setParameter("DS_MERCHANT_CURRENCY", moneda);
ApiMacSha256.setParameter("DS_MERCHANT_TRANSACTIONTYPE", trans);
ApiMacSha256.setParameter("DS_MERCHANT_TERMINAL", terminal);
ApiMacSha256.setParameter("DS_MERCHANT_MERCHANTURL", url);
ApiMacSha256.setParameter("DS_MERCHANT_URLOK", urlOK);
ApiMacSha256.setParameter("DS_MERCHANT_URLKO", urlKO);
```

Por último se debe llamar a la función de la librería "createMerchantParameters()", tal y como se muestra a continuación:

```
String params = ApiMacSha256.createMerchantParameters();
```

3. Calcular el parámetro **Ds_Signature**. Para llevar a cabo el cálculo de este parámetro, se debe llamar a la función de la librería "createMerchantSignature()" con la clave obtenida del módulo de administración, tal y como se muestra a continuación:

```
String claveModuloAdmin = "Mk9m98IfEblmPfrpsawt7BmxObt98Jev";
String signature = ApiMacSha256.createMerchantSignature(claveModuloAdmin);
```

4. Una vez obtenidos los valores de los parámetros **Ds_MerchantParameters** y **Ds_Signature** , se debe rellenar el formulario de pago con los valores obtenidos, tal y como se muestra a continuación:

```
<form action="https://sis.redsys.es/sis/realizarPago"
    method="POST" target="_blank">

    <input type="text" name="Ds_SignatureVersion"
        value="HMAC_SHA256_V1" />
    <input type="text" name="Ds_MerchantParameters"
        value="<%= params %>" />
    <input type="text" name="Ds_Signature"
    value="<%= signature %>" />
    <input type="submit" value="Realizar Pago" />

</form>
```

5. Recepción de la notificación on-line

La notificación on-line es una función opcional que permite a la tienda web recibir el resultado de una transacción de forma on-line y en tiempo real, una vez que el cliente ha completado el proceso en el TPV Virtual.

El comercio debe capturar **y validar todos los parámetros junto a la firma** de la notificación on-line de forma previa a cualquier ejecución en su servidor.

La utilización de las librerías de ayuda proporcionadas por Redsys se expone en los siguientes subapartados y dependerá del tipo de notificación configurada:

5.1 Notificación Síncrona y Asíncrona

En los apartados anteriores se ha descrito la forma de acceso al SIS utilizando conexión por Redirección y el sistema de firma basado en HMAC SHA256. En este apartado se explica cómo se utilizan las librerías disponibles PHP y JAVA para facilitar los desarrollos **para la recepción de los parámetros de la notificación on-line y la validación de la firma**. El uso de las librerías suministradas por Redsys es opcional, si bien simplifican los desarrollos a realizar por el comercio.

5.1.1 Librería PHP

A continuación se presentan los pasos que debe seguir un comercio para la utilización de la librería PHP proporcionada por Redsys:

1. Importar el fichero principal de la librería, tal y como se muestra a continuación:

```
include_once 'redsysHMAC256_API_PHP_4.0.2/apiRedsys.php';
```

   El comercio debe decidir si la importación desea hacerla con la función "include" o "required", según los desarrollos realizados.

2. Definir un objeto de la clase principal de la librería, tal y como se muestra a continuación:

```
$miObj = new RedsysAPI;
```

3. Capturar los parámetros de la notificación on-line:

```
$version = $_POST["Ds_SignatureVersion"];
$params = $_POST["Ds_MerchantParameters"];
$signatureRecibida = $_POST["Ds_Signature"];
```

4. Decodificar el parámetro **Ds_MerchantParameters**. Para llevar a cabo la decodificación de este parámetro, se debe llamar a la función de la librería "decodeMerchantParameters()", tal y como se muestra a continuación:

```
$decodec = $miObj->decodeMerchantParameters($params);
```

Una vez se ha realizado la llamada a la función "decodeMerchantParameters()", se puede obtener el valor de cualquier parámetro que sea susceptible de incluirse en la notificación on-line (Anexo 2 del apartado Anexos del presente documento). Para llevar a cabo la obtención del valor de un parámetro se debe llamar a la función "getParameter()" de la librería con el nombre de parámetro, tal y como se muestra a continuación para obtener el código de respuesta:

```
$codigoRespuesta = $miObj->getParameter("Ds_Response");
```

NOTA IMPORTANTE: Para garantizar la seguridad y el origen de las notificaciones el comercio debe llevar a cabo la validación de la firma recibida y de todos los parámetros que se envían en la notificación.

5. Validar el parámetro **Ds_Signature**. Para llevar a cabo la validación de este parámetro se debe calcular la firma y compararla con el parámetro **Ds_Signature** capturado. Para ello se debe llamar a la función de la librería "createMerchantSignatureNotif()" con la clave obtenida del módulo de administración y el parámetro **Ds_MerchantParameters** capturado, tal y como se muestra a continuación:

```
$claveModuloAdmin = 'Mk9m98IfEblmPfrpsawt7BmxObt98Jev';
$signatureCalculada = $miObj->createMerchantSignatureNotif($claveModuloAdmin,
                                                           $params);
```

Una vez hecho esto, ya se puede validar si el valor de la firma enviada coincide con el valor de la firma calculada, tal y como se muestra a continuación:

```
if ($signatureCalculada === $signatureRecibida){
    echo "FIRMA OK. Realizar tareas en el servidor";
} else {
    echo "FIRMA KO. Error, firma inválida";
}
```

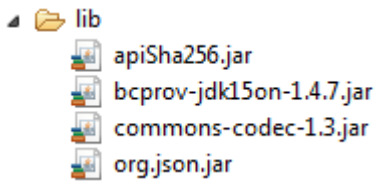
5.1.2 Librería JAVA

A continuación se presentan los pasos que debe seguir un comercio para la utilización de la librería JAVA proporcionada por Redsys:

1. Importar la librería, tal y como se muestra a continuación:

```
<%@page import="sis.redsys.api.ApiMacSha256"%>
```

El comercio debe incluir en la vía de construcción del proyecto todas las librerías(JARs) que se proporcionan:



2. Capturar los parámetros de la notificación on-line:

```
String version = request.getParameter("Ds_SignatureVersion");
String params = request.getParameter("Ds_MerchantParameters");
String signatureRecibida = request.getParameter("Ds_Signature");
```

3. Decodificar el parámetro **Ds_MerchantParameters**. Para llevar a cabo la decodificación de este parámetro, se debe llamar a la función de la librería "decodeMerchantParameters()", tal y como se muestra a continuación:

```
String decodec = ApiMacSha256.decodeMerchantParameters(params);
```

Una vez se ha realizado la llamada a la función "decodeMerchantParameters()", se puede obtener el valor de cualquier parámetro que sea susceptible de incluirse en la notificación on-line (Anexo 2 del apartado Anexos del presente documento). Para llevar a cabo la obtención del valor de un parámetro se debe llamar a la función "getParameter()" de la librería con el nombre de parámetro, tal y como se muestra a continuación para obtener el código de respuesta:

```
String codigoRespuesta = ApiMacSha256.getParameter("Ds_Response");
```

NOTA IMPORTANTE: Para garantizar la seguridad y el origen de las notificaciones el comercio debe llevar a cabo la validación de la firma recibida y de todos los parámetros que se envían en la notificación.

4. Validar el parámetro **Ds_Signature**. Para llevar a cabo la validación de este parámetro se debe calcular la firma y compararla con el parámetro **Ds_Signature** capturado. Para ello se debe llamar a la función de la librería "createMerchantSignatureNotif()" con la clave obtenida del módulo de administración y el parámetro **Ds_MerchantParameters** capturado, tal y como se muestra a continuación:

```
String claveModuloAdmin = "Mk9m98IfEblmPfrpsawt7BmxObt98Jev";
String signatureCalculada = ApiMacSha256.createMerchantSignatureNotif(claveModuloAdmin,
                                                                      params);
```

Una vez hecho esto, ya se puede validar si el valor de la firma enviada coincide con el valor de la firma calculada, tal y como se muestra a continuación:

```
if (signatureCalculada.equals(signatureRecibida)) {
    System.out.println("FIRMA OK. Realizar tareas en el servidor");
} else {
    System.out.println("FIRMA KO. Error, firma inválida");
}
```

5.2 Notificación SOAP

Las características del servicio SOAP que deben publicar los comercios se describe en el Anexo 3(Notificación SOAP) del apartado Anexos del presente documento.

En este apartado se explica cómo se utilizan las librerías disponibles PHP y JAVA para facilitar los desarrollos para la recepción de los parámetros de la notificación on-line(SOAP) y la validación de la firma. El uso de las librerías suministradas por Redsys es opcional, si bien simplifican los desarrollos a realizar por el comercio.

5.2.1 Librería PHP

A continuación se presentan los pasos que debe seguir un comercio para la utilización de la librería PHP proporcionada por Redsys:

1. Importar el fichero principal de la librería, tal y como se muestra a continuación:

```
include_once 'redsysHMAC256_API_PHP_4.0.2/apiRedsys.php';
```

El comercio debe decidir si la importación desea hacerla con la función "include" o "required", según los desarrollos realizados.

2. Definir un objeto de la clase principal de la librería, tal y como se muestra a continuación:

```
$miObj = new RedsysAPI;
```

3. Validar la firma que se envía en la notificación. Para llevar a cabo la validación de este parámetro se debe calcular la firma y compararla con la firma que se envía en la notificación. Para realizar el cálculo de la firma se debe llamar a la función de la librería "createMerchantSignatureNotifSOAPRequest()" con la clave obtenida del módulo de administración y el valor del mensaje recibido en la notificación.

```
function procesaNotificacionSIS($XML) {

$claveModuloAdmin = 'Mk9m98IfEblmPfrpsawt7BmxObt98Jev';
$signatureCalculada = $miObj->createMerchantSignatureNotifSOAPRequest($claveModuloAdmin,$XML);
```

Una vez hecho esto, el comercio debe capturar el valor de la firma recibida (parámetro **<Signature>**) y validar si el valor de esta coincide con el valor de la firma calculada, tal y como se muestra a continuación:

```
if ($signatureCalculada === $signatureRecibida){
    echo "FIRMA OK. Realizar tareas en el servidor";
} else {
    echo "FIRMA KO. Error, firma inválida";
}
```

NOTA IMPORTANTE: Para garantizar la seguridad y el origen de las notificaciones el comercio debe llevar a cabo la validación de la firma recibida y de todos los parámetros que se envían en la notificación.

4. Una vez validada la firma, el comercio debe enviar la respuesta de la notificación. Esta respuesta está firmada y para llevar a cabo el cálculo de la firma primero se debe capturar el número de pedido del mensaje recibido en la notificación. Para obtener el número de pedido se debe llamar a la función de la librería "getOrderNotifSOAP()" con el valor del mensaje recibido en la notificación.
   Una vez obtenido el número de pedido, tan sólo falta calcular la firma que se enviará en la respuesta. Para realizar el cálculo de la firma se debe llamar a la función de la librería "createMerchantSignatureNotifSOAPResponse()" con la clave obtenida del módulo de administración, el valor del mensaje de respuesta y el número de pedido capturado, tal y como se muestra a continuación:

```
$numPedido = $miObj->getOrderNotifSOAP($XML);

$response='<Response Ds_Version="0.0">
                <Ds_Response_Merchant>OK</Ds_Response_Merchant>
           </Response>';

$claveModuloAdmin = 'Mk9m98IfEblmPfrpsawt7BmxObt98Jev';

$firmaRespuesta = $miObj->createMerchantSignatureNotifSOAPResponse($claveModuloAdmin,
                                                                   $response,
                                                                   $numPedido);
```

Por último se debe formar el mensaje final mediante el mensaje de respuesta y la firma obtenida, tal y como se describe en el Anexo 3(Notificación SOAP) del apartado Anexos del presente documento.

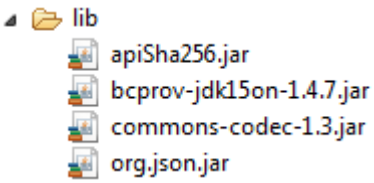
5.2.2 Librería JAVA

A continuación se presentan los pasos que debe seguir un comercio para la utilización de la librería JAVA proporcionada por Redsys:

1. Importar la librería, tal y como se muestra a continuación:

```
<%@page import="sis.redsys.api.ApiMacSha256"%>
```

El comercio debe incluir en la vía de construcción del proyecto todas las librerías(JARs) que se proporcionan:



2. Validar la firma que se envía en la notificación. Para llevar a cabo la validación de este parámetro se debe calcular la firma y compararla con la firma que se envía en la notificación. Para realizar el cálculo de la firma se debe llamar a la función de la librería "createMerchantSignatureNotifSOAPRequest()" con la clave obtenida del módulo de administración y el valor del mensaje recibido en la notificación.

```
String claveModuloAdmin = "Mk9m98IfEblmPfrpsawt7BmxObt98Jev";
String signatureCalculada = ApiMacSha256.createMerchantSignatureNotifSOAPRequest(claveModuloAdmin,XML);
```

Una vez hecho esto, el comercio debe capturar el valor de la firma recibida (parámetro **<Signature>**) y validar si el valor de esta coincide con el valor de la firma calculada, tal y como se muestra a continuación:

```
if (signatureCalculada.equals(signatureRecibida)) {
    System.out.println("FIRMA OK. Realizar tareas en el servidor");
} else {
    System.out.println("FIRMA KO. Error, firma inválida");
}
```

NOTA IMPORTANTE: Para garantizar la seguridad y el origen de las notificaciones el comercio debe llevar a cabo la validación de la firma recibida y de todos los parámetros que se envían en la notificación.

3. Una vez validada la firma, el comercio debe enviar la respuesta de la notificación. Esta respuesta está firmada y para llevar a cabo el cálculo de la firma primero se debe capturar el número de pedido del mensaje recibido en la notificación. Para obtener el número de pedido se debe llamar a la función de la librería "getOrderNotifSOAP()" con el valor del mensaje recibido en la notificación.
   Una vez obtenido el número de pedido, tan sólo falta calcular la firma que se enviará en la respuesta. Para realizar el cálculo de la firma se debe llamar a la función de la librería "createMerchantSignatureNotifSOAPResponse()" con la clave obtenida del módulo de administración, el valor del mensaje de respuesta y el número de pedido capturado, tal y como se muestra a continuación:

```
String numPedido = ApiMacSha256.getOrderNotifSOAP(XML);

String respons = "<Response Ds_Version='0.0'><Ds_Response_Merchant>OK</Ds_Response_Merchant></Response>";

String claveModuloAdmin = "Mk9m98IfEblmPfrpsawt7BmxObt98Jev";

String signatureCalculada = ApiMacSha256.createMerchantSignatureNotifSOAPResponse(claveModuloAdmin,
                                                                                   respons,
                                                                                   numPedido);
```

Por último se debe formar el mensaje final mediante el mensaje de respuesta y la firma obtenida, tal y como se describe en el Anexo 3(Notificación SOAP) del apartado Anexos del presente documento.

6. Retorno del control de la navegación

Una vez que el cliente ha realizado el proceso en el TPV Virtual, se redirige la navegación hacia a la tienda web. Este retorno a la web de la tienda se realiza hacia la URL comunicada como parámetro en la llamada inicial al TPV Virtual o en su defecto, se obtiene de la configuración del terminal en el módulo de administración del TPV Virtual. Se pueden disponer de URLs de retorno distintas según el resultado de la transacción (URL OK y URL KO).

El comercio debe capturar y validar, en caso de que la configuración de su comercio así lo requiera (Parámetro en las URLs = SI), los parámetros del retorno de control de navegación previo a cualquier ejecución en su servidor.

La utilización de las librerías de ayuda proporcionadas por Redsys para la captura y validación de los parámetros del retorno de control de navegación, se expone a continuación.

6.1 Utilización de librerías de ayuda

Una vez expuesta la nueva forma de acceso al SIS utilizando el sistema de firma basado en HMAC SHA256, este subapartado explica cómo se utilizan las librerías PHP y JAVA para la recepción de los parámetros del retorno de control de navegación.

6.1.1 Librería PHP

A continuación se presentan los pasos que debe seguir un comercio para la utilización de la librería PHP proporcionada por Redsys:

1. Importar el fichero principal de la librería, tal y como se muestra a continuación:

```
include_once 'redsysHMAC256_API_PHP_4.0.2/apiRedsys.php';
```

   El comercio debe decidir si la importación desea hacerla con la función "include" o "required", según los desarrollos realizados.

2. Definir un objeto de la clase principal de la librería, tal y como se muestra a continuación:

```
$miObj = new RedsysAPI;
```

3. Capturar los parámetros de la notificación on-line:

```
$version = $_GET["Ds_SignatureVersion"];
$params = $_GET["Ds_MerchantParameters"];
$signatureRecibida = $_GET["Ds_Signature"];
```

4. Decodificar el parámetro **Ds_MerchantParameters**. Para llevar a cabo la decodificación de este parámetro, se debe llamar a la función de la librería "decodeMerchantParameters()", tal y como se muestra a continuación:

```
$decodec = $miObj->decodeMerchantParameters($params);
```

Una vez se ha realizado la llamada a la función "decodeMerchantParameters()", se puede obtener el valor de cualquier parámetro que sea susceptible de incluirse en la notificación on-line (Anexo 2 del apartado Anexos del presente documento). Para llevar a cabo la obtención del valor de un parámetro se debe llamar a la función "getParameter()" de la librería con el nombre de parámetro, tal y como se muestra a continuación para obtener el código de respuesta:

```
$codigoRespuesta = $miObj->getParameter("Ds_Response");
```

NOTA IMPORTANTE: Es importante llevar a cabo la validación de todos los parámetros que se envían en la comunicación. Para actualizar el estado del pedido de forma on-line NO debe usarse esta comunicación, sino la notificacion on-line descrita en los otros apartados, ya que el retorno de la navegación depende de las acciones del cliente en su navegador.

5. Validar el parámetro **Ds_Signature**. Para llevar a cabo la validación de este parámetro se debe calcular la firma y compararla con el parámetro **Ds_Signature** capturado. Para ello se debe llamar a la función de la librería "createMerchantSignatureNotif()" con la clave obtenida del módulo de administración y el parámetro **Ds_MerchantParameters** capturado, tal y como se muestra a continuación:

```
$claveModuloAdmin = 'Mk9m98IfEblmPfrpsawt7BmxObt98Jev';
$signatureCalculada = $miObj->createMerchantSignatureNotif($claveModuloAdmin,
                                                           $params);
```

Una vez hecho esto, ya se puede validar si el valor de la firma enviada coincide con el valor de la firma calculada, tal y como se muestra a continuación:

```
if ($signatureCalculada === $signatureRecibida){
    echo "FIRMA OK. Realizar tareas en el servidor";
} else {
    echo "FIRMA KO. Error, firma inválida";
}
```

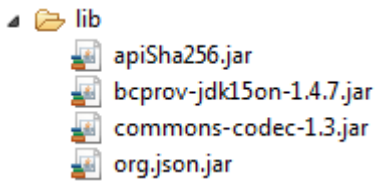
6.1.2 Librería JAVA

A continuación se presentan los pasos que debe seguir un comercio para la utilización de la librería JAVA proporcionada por Redsys:

1. Importar la librería, tal y como se muestra a continuación:

```
<%@page import="sis.redsys.api.ApiMacSha256"%>
```

El comercio debe incluir en la vía de construcción del proyecto todas las librerías(JARs) que se proporcionan:



2. Capturar los parámetros del retorno de control de navegación:

```
String version = request.getParameter("Ds_SignatureVersion");
String params = request.getParameter("Ds_MerchantParameters");
String signatureRecibida = request.getParameter("Ds_Signature");
```

3. Decodificar el parámetro **Ds_MerchantParameters**. Para llevar a cabo la decodificación de este parámetro, se debe llamar a la función de la librería "decodeMerchantParameters()", tal y como se muestra a continuación:

```
String decodec = ApiMacSha256.decodeMerchantParameters(params);
```

Una vez se ha realizado la llamada a la función "decodeMerchantParameters()", se puede obtener el valor de cualquier parámetro que sea susceptible de incluirse en la retorno de control de navegación (Anexo 2 del apartado Anexos del presente documento). Para llevar a cabo la obtención del valor de un parámetro se debe llamar a la función "getParameter()" de la librería con el nombre de parámetro, tal y como se muestra a continuación para obtener el código de respuesta:

```
String codigoRespuesta = ApiMacSha256.getParameter("Ds_Response");
```

NOTA IMPORTANTE: Es importante llevar a cabo la validación de todos los parámetros que se envían en la comunicación. Para actualizar el estado del pedido de forma on-line NO debe usarse esta comunicación, sino la notificación on-line descrita en los otros apartados, ya que el retorno de la navegación depende de las acciones del cliente en su navegador.

4. Validar el parámetro **Ds_Signature**. Para llevar a cabo la validación de este parámetro se debe calcular la firma y compararla con el parámetro **Ds_Signature** capturado. Para ello se debe llamar a la función de la librería "createMerchantSignatureNotif()" con la clave obtenida del módulo de administración y el parámetro **Ds_MerchantParameters** capturado, tal y como se muestra a continuación:

```
String claveModuloAdmin = "Mk9m98IfEblmPfrpsawt7BmxObt98Jev";
String signatureCalculada = ApiMacSha256.createMerchantSignatureNotif(claveModuloAdmin,
                                                                      params);
```

Una vez hecho esto, ya se puede validar si el valor de la firma enviada coincide con el valor de la firma calculada, tal y como se muestra a continuación:

```
if (signatureCalculada.equals(signatureRecibida)) {
    System.out.println("FIRMA OK. Realizar tareas en el servidor");
} else {
    System.out.println("FIRMA KO. Error, firma inválida");
}
```

7. Códigos de error asociados

Se han definido nuevos códigos de error propios de este nuevo acceso al SIS:

SIS0429	Error en la versión enviada por el comercio en el parámetro **Ds_SignatureVersion**
SIS0430	Error al decodificar el parámetro **Ds_MerchantParameters**
SIS0431	Error del objeto JSON que se envía codificado en el parámetro **Ds_MerchantParameters**
SIS0432	Error FUC del comercio erróneo
SIS0433	Error Terminal del comercio erróneo
SIS0434	Error ausencia de número de pedido en la operación enviada por el comercio
SIS0435	Error en el cálculo de la firma

Además de los errores propios de este nuevo acceso, se deben tener en cuenta los errores ya recogidos en la "Guía de Comercios" del SIS.

8. ANEXOS

8.1 Datos de la solicitud de pago

En la petición de pago se tendrán que enviar una serie de datos obligatorios. El comercio también podrá incluir datos adicionales según las necesidades de su operativa concreta.

Los datos imprescindibles para la gestión de la autorización están marcados como obligatorios en la tabla siguiente.

DATO	NOMBRE DEL DATO	Long. / Tipo	COMENTARIOS
Identificación de comercio: código FUC	*Ds_Merchant_MerchantCode*	*9/N.*	**Obligatorio**. Código FUC asignado al comercio.
Número de terminal	*Ds_Merchant_Terminal*	*3/N.*	**Obligatorio**. Número de terminal que le asignará su banco. Tres se considera su longitud máxima
Tipo de transacción	*Ds_Merchant_TransactionType*	*1 / Num*	**Obligatorio**. para el comercio para indicar qué tipo de transacción es. Los posibles valores son: 0 – **Autorización** 1 – **Preautorización** 2 – **Confirmación de preautorización** 3 – **Devolución Automática** 5 – **Transacción Recurrente** 6 – **Transacción Sucesiva** 7 – **Pre-autenticación** 8 – **Confirmación de pre-autenticación** 9 – **Anulación de Preautorización** O **– Autorización en diferido** P– **Confirmación de autorización en diferido** Q **- Anulación de autorización en diferido** R **– Cuota inicial diferido** S **– Cuota sucesiva diferido**
Importe	*Ds_Merchant_Amount*	*12 / Núm.*	**Obligatorio**. Para Euros las dos últimas posiciones se consideran decimales.
Moneda	*Ds_Merchant_Currency*	*4 / Núm.*	**Obligatorio**. Se debe enviar el código numérico de la moneda según el ISO-4217, por ejemplo: 978 euros 840 dólares 826 libras 392 yenes 4 se considera su longitud máxima
Número de Pedido	*Ds_Merchant_Order*	*12 / A-N.*	**Obligatorio**. Los 4 primeros dígitos deben ser numéricos, para los dígitos restantes solo utilizar los siguientes caracteres ASCII Del 30 = **0** al 39 = **9** Del 65 = **A** al 90 = **Z** Del 97 = **a** al 122 = **z**
URL del comercio para la notificación "on-line"	*Ds_Merchant_MerchantURL*	*250/A-N*	Obligatorio **si** el comercio tiene notificación "on-line". URL del comercio que recibirá un post con los datos de la transacción.

DATO	NOMBRE DEL DATO	Long. / Tipo	COMENTARIOS
Descripción del producto	*Ds_Merchant_ProductDescription*	*125 / A-N*	Opcional. 125 se considera su longitud máxima. Este campo se mostrará al titular en la pantalla de confirmación de la compra.
Nombre y apellidos del titular	*Ds_Merchant_Titular*	*60/A-N*	Opcional. Su longitud máxima es de 60 caracteres. Este campo se mostrará al titular en la pantalla de confirmación de la compra.
URLOK	*Ds_Merchant_UrlOK*	*250/A-N*	Opcional: si se envía será utilizado como URLOK ignorando el configurado en el módulo de administración en caso de tenerlo.
URL KO	*Ds_Merchant_UrlKO*	*250/A-N*	Opcional: si se envía será utilizado como URLKO ignorando el configurado en el módulo de administración en caso de tenerlo
Identificación de comercio: denominación comercial	*Ds_Merchant_MerchantName*	*25/A-N*	Opcional: será el nombre del comercio que aparecerá en el ticket del cliente (opcional).
Idioma del titular	*Ds_Merchant_ConsumerLanguage*	*3/N.*	Opcional: el Valor 0, indicará que no se ha determinado el idioma del cliente (opcional). Otros valores posibles son: Castellano-001, Inglés-002, Catalán-003, Francés-004, Alemán-005, Holandés-006, Italiano-007, Sueco-008, Portugués-009, Valenciano-010, Polaco-011, Gallego-012 y Euskera-013.
Importe total (cuota recurrente)	Ds_Merchant_SumTotal	12/N.	Obligatorio. Representa la suma total de los importes de las cuotas. Las dos últimas posiciones se consideran decimales.
Datos del comercio	*Ds_Merchant_MerchantData*	*1024 /A-N*	Opcional para el comercio para ser incluidos en los datos enviados por la respuesta "on-line" al comercio si se ha elegido esta opción.
Frecuencia	*Ds_Merchant_DateFrecuency*	*5/ N*	Frecuencia en días para las transacciones recurrentes y recurrentes diferidas (obligatorio para recurrentes)
Fecha límite	*Ds_Merchant_ChargeExpiryDate*	*10/ A-N*	Formato yyyy-MM-dd fecha límite para las transacciones Recurrentes (Obligatorio para recurrentes y recurrentes diferidas)
Código de Autorización	*Ds_Merchant_AuthorisationCode*	*6 / Num*	Opcional. Representa el código de autorización necesario para identificar una transacción recurrente sucesiva en las devoluciones de operaciones recurrentes sucesivas. Obligatorio en devoluciones de operaciones recurrentes.
Fecha de la operación recurrente sucesiva	*Ds_Merchant_TransactionDate*	*10 / A-N*	Opcional. Formato yyyy-mm-dd. Representa la fecha de la cuota sucesiva, necesaria para identificar la transacción en las devoluciones. Obligatorio en las devoluciones de cuotas sucesivas y de cuotas sucesivas diferidas.

8.2 Datos de la notificación on-line

Recomendamos el uso de este método, ya que permite que la tienda web reciba el resultado de las transacciones, de forma on-line en tiempo real. La Notificación ON-LINE es configurable en el módulo de administración, y admite varías posibilidades en función de la necesidad del comercio. Tanto la notificación HTTP como la notificación por mail tienen exactamente el mismo formato.

La notificación por HTTP envía al comercio un POST independiente de la conexión con el navegador del comprador, y no tiene ningún reflejo en pantalla del mismo. Evidentemente, en el lado del comercio, deberá haber un proceso que recoja esta respuesta. Para ello tendrá que facilitar una URL donde recibir estas respuestas en el formulario web que envía al realizar la solicitud de autorización (ver el campo Ds_Merchant_MerchantURL en "Datos del formulario de pago"). Esta URL será un CGI, Servlet, etc. desarrollado en el lenguaje que el comercio considere adecuado para integrar en su Servidor (C, Java, Perl, PHP, ASP, etc.), capaz de interpretar la respuesta que le envíe el TPV Virtual.

NOTA: Estos mismos datos se incorporarán en la URL OK (Ds_Merchant_UrlOK) o URL KO (Ds_Merchant_UrlKO) si el comercio tiene activado el envío de parámetros en la redirección de respuesta.

DATO	NOMBRE DEL DATO	LONG/TIPO	COMENTARIOS
Fecha	*Ds_Date*	*dd/mm/yyyy*	Fecha de la transacción
Hora	*Ds_Hour*	*HH:mm*	Hora de la transacción
Importe	*Ds_Amount*	*12 / Núm.*	Mismo valor que en la petición.
Moneda	*Ds_Currency*	*4 / Núm.*	Mismo valor que en la petición. 4 se considera su longitud máxima.
Número de pedido	*Ds_Order*	*12 / A-N.*	Mismo valor que en la petición.
Identificación de comercio: código FUC	*Ds_MerchantCode*	*9 / N.*	Mismo valor que en la petición.
Terminal	*Ds_Terminal*	*3 / Núm.*	Número de terminal que le asignará su banco. 3 se considera su longitud máxima.
Código de respuesta	*Ds_Response*	*4 / Núm.*	Ver tabla siguiente (Posibles valores del Ds_Response).
Datos del comercio	*Ds_MerchantData*	*1024 / A-N*	Información opcional enviada por el comercio en el formulario de pago.
Pago Seguro	*Ds_SecurePayment*	*1 / Núm.*	0 – Si el pago **NO** es seguro 1 – Si el pago es seguro
Tipo de operación	*Ds_TransactionType*	*1 / A-N*	Tipo de operación que se envió en el formulario de pago

DATO	NOMBRE DEL DATO	LONG/TIPO	COMENTARIOS
País del titular	Ds_Card_Country	3/Núm	Opcional: País de emisión de la tarjeta con la que se ha intentado realizar el pago. En el siguiente enlace es posible consultar los códigos de país y su correspondencia: http://unstats.un.org/unsd/methods/m49/m49alpha.htm
Código de autorización	Ds_AuthorisationCode	6/ A-N	Opcional: Código alfanumérico de autorización asignado a la aprobación de la transacción por la institución autorizadora.
Idioma del titular	Ds_ConsumerLanguage	3 / Núm	Opcional: El valor 0, indicará que no se ha determinado el idioma del cliente. (opcional). 3 se considera su longitud máxima.
Tipo de Tarjeta	Ds_Card_Type	1 / A-N	Opcional: Valores posibles: C – Crédito D - Débito

Estos son los posibles valores del Ds_Response o "Código de respuesta":

CÓDIGO	SIGNIFICADO
101	Tarjeta caducada
102	Tarjeta en excepción transitoria o bajo sospecha de fraude
106	Intentos de PIN excedidos
125	Tarjeta no efectiva
129	Código de seguridad (CVV2/CVC2) incorrecto
180	Tarjeta ajena al servicio
184	Error en la autenticación del titular
190	Denegación del emisor sin especificar motivo
191	Fecha de caducidad errónea
202	Tarjeta en excepción transitoria o bajo sospecha de fraude con retirada de tarjeta
904	Comercio no registrado en FUC
909	Error de sistema
913	Pedido repetido
944	Sesión Incorrecta
950	Operación de devolución no permitida
9912/912	Emisor no disponible
9064	Número de posiciones de la tarjeta incorrecto
9078	Tipo de operación no permitida para esa tarjeta
9093	Tarjeta no existente
9094	Rechazo servidores internacionales
9104	Comercio con "titular seguro" y titular sin clave de compra segura
9218	El comercio no permite op. seguras por entrada /operaciones

9253	Tarjeta no cumple el check-digit
9256	El comercio no puede realizar preautorizaciones
9257	Esta tarjeta no permite operativa de preautorizaciones
9261	Operación detenida por superar el control de restricciones en la entrada al SIS
9913	Error en la confirmación que el comercio envía al TPV Virtual (solo aplicable en la opción de sincronización SOAP)
9914	Confirmación "KO" del comercio (solo aplicable en la opción de sincronización SOAP)
9915	A petición del usuario se ha cancelado el pago
9928	Anulación de autorización en diferido realizada por el SIS (proceso batch)
9929	Anulación de autorización en diferido realizada por el comercio
9997	Se está procesando otra transacción en SIS con la misma tarjeta
9998	Operación en proceso de solicitud de datos de tarjeta
9999	Operación que ha sido redirigida al emisor a autenticar

Estos códigos de respuesta se muestran en el campo "Código de respuesta" de la consulta de operaciones, siempre y cuando la operación no está autorizada, tal y como se muestra en la siguiente imagen:

8.3 Notificación SOAP

El servicio SOAP que deben publicar los comercios debe tener las siguientes características:

1. El servicio deberá llamarse 'InotificacionSIS' y ofrecer un método llamado 'procesaNotificacionSIS'. Este método estará definido con un parámetro de entrada tipo cadena XML y otro parámetro de salida del mismo tipo. Para más información, se adjunta un fichero WSDL a partir del cual se puede construir el esqueleto del servidor y que servirá para definir los tipos de datos que se intercambiarán entre cliente y servidor, de cara a facilitar la comunicación.
2. El formato de los mensajes que se intercambiarán en este servicio deberán ajustarse a la siguiente dtd:
3. Mensaje de notificación enviado desde el SIS con los datos de la operación correspondiente:

```
<!ELEMENT Message (Request, Signature)>

<!ELEMENT Request (Fecha, Hora, Ds_SecurePayment, Ds_Amount, Ds_Currency, Ds_Order,
Ds_MerchantCode, Ds_Terminal, Ds_Response, Ds_MerchantData?, Ds_Card_Type?,
DS_Card_Type?, Ds_TransactionType, Ds_ConsumerLanguage, Ds_ErrorCode?,
Ds_CardCountry?, Ds_AuthorisationCode?)>

<!ATTLIST Request Ds_Version CDATA #REQUIRED>
<!ELEMENT Fecha (#PCDATA)>
<!ELEMENT Hora (#PCDATA)>
<!ELEMENT Ds_SecurePayment (#PCDATA)>
<!ELEMENT Ds_Amount (#PCDATA)>
<!ELEMENT Ds_Currency (#PCDATA)>
<!ELEMENT Ds_Order (#PCDATA)>
<!ELEMENT Ds_MerchantCode (#PCDATA)>
<!ELEMENT Ds_Terminal (#PCDATA)>
<!ELEMENT Ds_Response (#PCDATA)>
<!ELEMENT Ds_MerchantData (#PCDATA)>
<!ELEMENT Ds_Card_Type (#PCDATA)>
<!ELEMENT Ds_TransactionType (#PCDATA)>
<!ELEMENT Ds_ConsumerLanguage (#PCDATA)>
<!ELEMENT Ds_ErrorCode (#PCDATA)>
<!ELEMENT Ds_CardCountry (#PCDATA)>
<!ELEMENT Ds_AuthorisationCode (#PCDATA)>
<!ELEMENT Signature (#PCDATA)>
<!ELEMENT DS_Card_Type (#PCDATA)>
```

Para generar el valor del campo Signature en el mensaje de respuesta del comercio aplicaremos un HMAC SHA-256 de la cadena <Request ...>…</Request>.

Ejemplo:

Sea el siguiente mensaje:

```
<Message>
  <Request Ds_Version="0.0">
        <Fecha>01/04/2003</Fecha>
        <Hora>16:57</Hora>
        <Ds_SecurePayment>1</Ds_SecurePayment>
        <Ds_Amount>345</Ds_Amount>
        <Ds_Currency>978</Ds_Currency>
        <Ds_Order>165446</Ds_Order>
         <Ds_MerchantCode>999008881</Ds_MerchantCode>
          <Ds_Terminal>001</Ds_Terminal>
           <Ds_Card_Country>724</Ds_Card_Country>
        <Ds_Response>0000</Ds_Response>
        <Ds_MerchantData>Alfombrilla para raton</Ds_MerchantData>
          <Ds_Card_Type>C</Ds_Card_Type>
        <Ds_TransactionType>1</Ds_TransactionType>
        <Ds_ConsumerLanguage>1</Ds_ConsumerLanguage>
           </Request>
   </Message>
```

Mensaje de respuesta del comercio a la notificación:

Ejemplo:

```
<!ELEMENT Message (Response, Signature)>
<!ELEMENT Response (Ds_Response_Merchant)>
<!ATTLIST Response Ds_Version CDATA #REQUIRED>
<!ELEMENT Ds_Response_Merchant (#PCDATA)>
<!ELEMENT Signature (#PCDATA)>
```

Los posibles valores que podrá tomar la etiqueta Ds_Response_Merchant serán:

- 'OK' cuando la notificación se ha recibido correctamente.
- 'KO' cuando se ha producido algún error.

Para generar el valor del campo Signature en el mensaje de respuesta del comercio aplicaremos un HMAC SHA-256 de la cadena <Response>…</Response>.

- **Ejemplos de mensajes intercambiados en una notificación con Sincronización SOAP:**

Mensaje de notificación enviado desde el SIS:

```
<Message>
        <Request Ds_Version="0.0">
    <Fecha>01/04/2003</Fecha>
    <Hora>16:57</Hora>
    <Ds_SecurePayment>1</Ds_SecurePayment>
    <Ds_Amount>345</Ds_Amount>
    <Ds_Currency>978</Ds_Currency>
    <Ds_Order>165446</Ds_Order>
    <Ds_Card_Type>C</Ds_ Card_Type >
    <Ds_MerchantCode>999008881</Ds_MerchantCode>
      <Ds_Terminal>001</Ds_Terminal>
      <Ds_Card_Country>724</Ds_Card_Country>
    <Ds_Response>0000</Ds_Response>
    <Ds_MerchantData>Alfombrilla para raton</Ds_MerchantData>
    <Ds_TransactionType>1</Ds_TransactionType>
    <Ds_ConsumerLanguage>1</Ds_ConsumerLanguage>
</Request>
<Signature>I3gacbQMEvUYN59YiHkiml-crEMwFAeogI1jlLBDFiw=</Signature>
  </Message>
```

Mensaje de respuesta desde el comercio al SIS:

```
    <Message>
    <Response Ds_Version="0.0">
    <Ds_Response_Merchant>OK</Ds_Response_Merchant>
    </Response>
<Signature>d/VtqOzNlds9MTL/QO12TvGDNT+yTfawFlg55ZcjX9Q=</Signature>
    </Message>
```

WSDL para el servicio InotificacionSIS

Los comercios que deseen desarrollar un servicio SOAP deben ajustarse a esta WSDL. A partir de ella y, mediante herramientas de generación automática de código, se puede desarrollar el esqueleto del servidor SOAP de forma cómoda y rápida.

La WSDL que debe cumplir el servicio SOAP desarrollado por el cliente es la siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>

<definitions name="InotificacionSIS"
targetNamespace=https://sis.SERMEPA.es/sis/InotificacionSIS.wsdl
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:tns="https://sis.SERMEPA.es/sis/InotificacionSIS.wsdl"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns="http://schemas.xmlsoap.org/wsdl/">

<message name="procesaNotificacionSISRequest">
  <part name="XML" type="xs:string"/>
</message>

<message name="procesaNotificacionSISResponse">
  <part name="return" type="xs:string"/>
</message>

<portType name="InotificacionSISPortType">
  <operation name="procesaNotificacionSIS">
    <input message="tns:procesaNotificacionSISRequest"/>
    <output message="tns:procesaNotificacionSISResponse"/>
  </operation>
</portType>

<binding name="InotificacionSISBinding" type="tns:InotificacionSISPortType">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="procesaNotificacionSIS">
    <soap:operation
soapAction="urn:InotificacionSIS#procesaNotificacionSIS" style="rpc"/>
    <input>
      <soap:body use="encoded"
encodingStyle=http://schemas.xmlsoap.org/soap/encoding/ namespace="InotificacionSIS"/>
    </input>
    <output>
      <soap:body use="encoded"
encodingStyle=http://schemas.xmlsoap.org/soap/encoding/ namespace="InotificacionSIS"/>
    </output>
  </operation>
</binding>

<service name="InotificacionSISService">
  <port name="InotificacionSIS" binding="tns:InotificacionSISBinding">
    <soap:address location="http://localhost/WebServiceSIS/InotificacionSIS.asmx"/>
  </port>
</service>


    </definitions>
```