Tabla de contenido

1. OBJETIVO

Este documento explica el funcionamiento del SDK-RUNT (Software Development Kit – RUNT) para la consulta de vehículos por servicio REST (Representational State Transfer) por las aseguradoras.

2. CONSULTA DE ASEGURADORAS - AMBIENTE PRODUCCIÓN

2.1 Requisitos

Para poder realizar consultas ante el RUNT en ambiente de producción cada aseguradora deberá contar con:

1. Un certificado digital de persona jurídica, con su respectiva llave. Que debe cumplir con los requisitos descritos en la sección 4. REQUISITOS PARA CERTIFICADOS CONSULTA SERVICIOS REST (Representational State Transfer)
2. Una dirección IP única desde la cual realizará las consultas.
3. Un identificador de usuario que será provisto por el RUNT.
4. El certificado público del RUNT el cual será enviado por RUNT, Se debe validar el certificado una vez sea descargado.
5. Una dirección IP que será suministrada por el RUNT, para ser configurado dentro del SDK, en esta dirección se publicarán los servicios de consulta.

2.2 Modelo de seguridad

El modelo de seguridad está soportado por firmas digitales, códigos de autenticación de mensajes (HMAC) y envío de información sobre canales seguros (SSL).

El procedimiento en general negocia una llave simétrica para intercambio de datos, mediante una petición firmada con la llave asimétrica (Emitida por autoridad de certificación) todo mediante un canal seguro, con esta llave simétrica se genera un HMAC y se envía en cada petición. A continuación se describe el procedimiento de manera más detallada.

1. El cliente establece una sesión SSL con el Servicio, mediante el uso de certificados digitales válidos.
2. El servicio valida que la IP del cliente esté debidamente autorizada.
3. Se genera una petición para negocio de llave simétrica y esta petición la firma el cliente utilizando el certificado digital de persona jurídica válido.
4. Cada petición que realiza el cliente Rest (Representational State Transfer), debe estar acompañado de un mensaje HMAC para su validación por el servicio.
5. La petición es validada, tanto en su forma como en su integridad (mediante el HMAC) en el RUNT y si la verificación es válida, la petición se procesa; de otra manera se rechaza.
6. La llave simétrica se puede renegociar en cualquier momento, por parte del cliente
7. Toda solicitud de negocio de clave queda registrado en una auditoría.
8. El número máximo de peticiones por unidad de tiempo, se configura para cada aseguradora.

2.3 Uso del SDK (Software Development Kit – RUNT)

Como parte del servicio de consulta el RUNT provee un Software Development Kit (SDK-RUNT) que facilita la integración de los servicios en el ambiente de nuestros clientes. Este SDK permite configurar fácilmente un cliente e implementa el protocolo de comunicación definido por el RUNT para el consumo de los servicios. Este SDK ha sido diseñado para lenguaje JAVA en versiones 6 o superiores. Para integrar el SDK-RUNT en su aplicación se debe:

1. Integrar en el proyecto la librería runt-sdk-aseguradoras-X.X.X.jar, runt-api-aseguradoras-X.X.X (En donde XXX corresponde a la versión más recientemente liberada).
2. Configurar las librerías necesarias que requieren estos dos jars (Para el caso de un cliente standalone, se suministrar todas las librerías necesarias dentro del paquete)
3. Utilizar la clase co.com.runt.sdk.aseguradoras.impl. ClienteRuntBuilder para construir un cliente que realice la consulta de los servicio. A continuación se presenta un ejemplo del uso del API (*Tenga en cuenta deberá contar con **TODA** la información descrita en los Requisitos para poder realizar consultas*).

```
ClienteRuntBuilder builder = new ClienteRuntBuilder();
ClienteRunt clienteRunt = builder
    .idCliente("123")
    .servidor("IP-RUnt-")
    .llavePublicaRunt(llavePublicaRunt)
    .llaveFirmaDigital(llaveFirmaDigital)
    .build();
```

4. La primera vez que se disponga a realizar consultas se debe negociar una llave privada con el servidor. Utilizando esta llave se generará un HMAC para cada uno de los mensajes intercambiados, este HMAC permitirá validar la autenticidad e integridad de los mensajes. Para generar una nueva llave se debe invocar el método *negociarLlaveConsulta,* el cual realizará la negociación con el servicio y retornará la llave negociada.

```
String llaveConsulta = clienteRunt.negociarLlaveConsulta();
```

Las llaves de consulta no tienen fecha de vencimiento, se recomienda que almacene esta llave en un ambiente seguro y la utilice posteriormente para realizar las consultas. Una llave nueva podrá ser negociada en cualquier momento utilizando su firma digital. Solo una llave de consulta estará vigente en cualquier momento, por lo que la negociación de una llave invalida la llave anterior que se encuentre registrada en el sistema.

Cuando se cuenta con una llave privada de consulta en la construcción del cliente RUNT se debe invocar el método *llavePrivadaConsulta* de la clase ClienteRuntBuilder cuando se esté configurando el cliente como se muestra a continuación.

```
ClienteRuntBuilder builder = new ClienteRuntBuilder();
ClienteRunt clienteRunt = builder
    .idCliente("123")
    .servidor("localhost")
    .llavePrivadaConsulta("llavePrivadaConsulta")
    .llavePublicaRunt(llavePublicaRunt)
    .llaveFirmaDigital(llaveFirmaDigital)
    .build();
```

5. Una vez se tenga configurado un cliente se pueden invocar dos métodos de consulta consultarVehiculoPorPlaca y consultarVehiculoPorVin que consultan la información de un vehículo dada su placa y VIN respectivamente. Ver Javadoc del API para detalles de la información entregada por el servicio.
    a. La información que se entrega del vehículo por el servicio es la siguiente, con los respectivos tipos de datos es la siguiente:

La información que se entrega del vehículo, se encuentra agrupada en las siguientes secciones:

a. Información General del Vehículo

Id Línea.
Línea.
Id Marca.
Marca.
Modelo.
Número de Chasis.
Número Ejes.
Número Licencia Transito.
Número Motor.
Número Placa.
Número Registro.
Número Serie.
Número Vin.
Capacidad Carga.
Cilindraje.
Id Clase Vehículo.
Clase Vehículo.
Clasificación.
Id Color.
Color.
Id Tipo Servicio.
Tipo Servicio.
Estado Del Vehículo.
Numero Regrabación Chasis.
Numero Regrabación Motor.
Numero Regrabación Serie.
Numero Regrabación Vin.
Organismo Transito.
Divipola (corresponde al código de centro poblado DIVIPOLA de 8 dígitos)
Id País Origen.
País Origen.
Peso Bruto Vehicular.
Días Matriculado.
Es Regrabado Chasis.
Es Regrabado Motor.
Es Regrabado Serie.
Es Regrabado Vin.
Fecha Matricula.
Tiene Prendas(s/n)
Tiene Gravámenes (s/n)

En la versión runt-api-aseguradoras-1.0.10, se incluyeron los campos Id Marca, id línea, id marca, id Color, id país origen, id clase vehículo y código DIVIPOLA.

b. Información técnica del vehículo

Peso Bruto Vehicular
Número de Ejes
Alto
Ancho
Largo
Número Llantas
Capacidad Carga

c. Información de Blindaje del vehículo

Es Blindado
Nivel de Blindaje
Fecha de Blindaje

d. Información de Certificados de desintegración

Entidad Desintegradora.
Estado Certificado.
Fecha Expedición.
No Certificado.

e. Información de propietarios

Cantidad de propietarios
Tipo de propietarios

f. Respecto a la información de la RTM de vehículos la información que se suministra es:

Cda Expide.
Fecha Expedición.
Fecha Vigente.
Tipo Revisión.
Es Vigente.

g. Respecto a la información del SOAT para vehículos nacionales se exponen los siguientes datos:

Entidad Expide Soat.
Estado.
Fecha Expedición.
Fecha Vencimiento.
Fecha Vigencia.
Número Póliza.

h. Respecto a la información del SOAT para vehículos extranjeros se exponen los siguientes datos.

Cda Expide.
Entidad Expide Soat.
Estado.
Fecha Expedición.
Fecha Vencimiento.
Fecha Vigencia.
No Póliza.

i. Información tarjetas de operación se exponen los siguientes datos.

Empresa Afiliadora.
Estado.
Fecha Expedición.
Fecha Fin.
Fecha Inicio.
Modalidad Servicio.
Modalidad Transporte.
Radio Acción.

j. Información pólizas de Caución se exponen los siguientes datos.

Estado Certificado.
Estado Póliza.
Fecha Expedición.
Fecha Vigencia Póliza.
Número Certificación.
Número Póliza.

k. Respecto a la información de los certificados de la DIJIN se exponen los siguientes datos.

Entidad Certificado.
Estado Certificado.
Fecha Expedición.
Número Certificado.

l. Limitaciones a la propiedad

Tipo de limitación
Numero Documento
Entidad Jurídica
Tipo Documento Demandante
No Documento demandante
Fecha de Expedición
Fecha de radicación
Descripción

Información de Repotenciación
Indicador es repotenciado
Fecha de repotenciación
Modelo a que se repotenció.

Los demás datos están especificados en el formato Javadoc adjunto al presente documento.

Para realizar estas consultas, se expone el siguiente ejemplo del uso adecuado del API

```
package co.com.runt.sdk.aseguradoras.test;



import co.com.runt.sdk.aseguradora.api.Vehiculo;

import co.com.runt.sdk.aseguradoras.api.ClienteRunt;

import co.com.runt.sdk.aseguradoras.impl.ClienteRuntBuilder;

import java.io.FileInputStream;

import java.security.KeyStore;
import java.security.PrivateKey;
import java.security.cert.Certificate;
import java.security.cert.CertificateFactory;


/**

 *
```

```
 * @author Concesión RUNT

 */


class Main {


    public static void main(String[] params) throws Exception {
        final KeyStore store = KeyStore.getInstance("JKS");

      //Se lee la el store que contiene el certificado
      store.load(new
FileInputStream("C:\\Users\\Calvarez\\Documents\\proyectos\\web_service_c
onsulta_aseguradoras\\aseguradora_14465657\\aseguradora_14465657.jks"),
"123456789".toCharArray());
        CertificateFactory cf = CertificateFactory.getInstance("X.509");

//Se lee el certificado de llave pública del RUNT
 Certificate runtCertificate = cf.generateCertificate(new
FileInputStream("C:\\runt_key.cer"));
        final ClienteRuntBuilder builder = new ClienteRuntBuilder();

//Se obtiene la llave privada de la aseguradora
        final PrivateKey pk = (PrivateKey)
store.getKey("aseguradora_14465657", "123456789".toCharArray());

//Cliente para realización de consulta. El idCliete corresponde al NIT de
la aseguradora, servidor corresponde a la IP del servidor del RUNT,
  final ClienteRunt clienteRunt = builder
                .idCliente("14465657")
                .servidor("190.254.17.42")
                .llavePublicaRunt(runtCertificate.getPublicKey())
                .llaveFirmaDigital(pk)
// En caso que haber negociado previamente la llave se debe incluir
dentro de la petición
.llavePrivadaConsulta("BnzkyEsEYJCynPSwVrqjQ0cfVs3NaZxXsRuLgLQ/EMo=")
                .build();


//En caso de ser la primera vez,  de debe negociar una llave de consulta,
para luego ser utilizado, guardarla de manera segura para luego enviarla
en la creación del  clienteRunt
        String llaveConsulta = clienteRunt.negociarLlaveConsulta();
//Ahora se pueden realizar las consultas
     Vehiculo vehiculo = clienteRunt.consultarVehiculoPorPlaca("DDC416");
```

```
        Vehiculo vehiculoVin =
clienteRunt.consultarVehiculoPorVin("9F9LAP3N8CM087788s");
        System.out.println(vehiculo.toString());
        System.out.println(vehiculoVin.toString());
        clienteRunt.terminar();
    }

}
```

2.4 Uso del SDK RUNT, Ambiente de producción

En ambiente de producción el proceso de configuración es similar al ambiente de pruebas, sin embargo se debe tener en cuenta las siguientes indicaciones:

- El certificado digital de persona jurídica, no es suministrado por el RUNT, debe ser adquirido ante una autoridad certificado aprobada por el la SIC. El resguardo de la llave privada del mismo certificado es de única responsabilidad de la entidad que lo adquiere. El RUNT en ningún momento solicitará acceso a la llave privada de la entidad.
- El SDK suministrado por el RUNT requiere acceso a la llave privada del cliente para poder generar las firmas respectivas.
- EL SDK del RUNT en ningún momento expone funcionalidad que envíe o copie la llave privada del cliente.
- Los password de acceso a la llave privada de la entidad no son suministrados por el RUNT y nunca estarán en su posesión, solo estarán y serán responsabilidad de la entidad que los posee.

Teniendo en cuenta las anteriores aclaraciones, en la secciones de código en dónde se lee los keystores de los certificados se debe utilizar los certificados que la entidad ha comprado y cuyas contraseñas están en su poder solamente.

El cliente
```
final KeyStore store = KeyStore.getInstance("JKS");

     //Se lee la el store que contiene el certificado
     store.load(new FileInputStream("ARCHIVO.jks"),
"PASSWORD".toCharArray());
```

```
final PrivateKey pk = (PrivateKey) store.getKey("ALIAS,
"PASSWORD".toCharArray());
```

Por tanto los datos de ARCHIVO.JKS, ALIAS y PASSWORD, no pueden ser suministrados por el RUNT, porque solo deben estar en posesión de la entidad.

3. CONSULTA DE ASEGURADORAS - AMBIENTE DE PRUEBA

3.1 Elementos suministrados por el RUNT

Para poder realizar consultas ante el RUNT en ambiente de pruebas, el RUNT suministra los siguientes elementos:

1. Un almacén de certificados denominado aseguradora_#{NIT}.jks en formato jks o aseguradora_#{NIT}.p12 en formato p12. El password será suministrado en un correo electrónico posterior. Este almacén debe ser utilizado como se indica en la en el documento Manual Técnico para consulta de aseguradoras.
2. El identificador de usuario será el NIT de la empresa.
3. El certificado público del RUNT ambiente de pruebas que se encuentra en el archivo runt_key.cer.
4. La IP del servidor de pruebas que debe ser utilizado para inicializar el sdk, será suministrado en un correo electrónico posterior, una vez la entidad sea autorizada para el uso de servicios.
5. Los datos que se consulten en el ambiente de pruebas, son datos que no necesariamente corresponden a la realidad y solo se exponen con el fin de realizar pruebas del servicio.
6. La consulta en el ambiente de pruebas está limitada en número para garantizar la estabilidad.
7. La Entidad debe suministrar la IP desde donde se realizarán las consultas, para ser habilitada dentro del esquema de seguridad del RUNT.

4. REQUISITOS PARA CERTIFICADOS CONSULTA SERVICIOS REST (Representational State Transfer)

Requisitos mínimos certificado digitales que se utilicen dentro para la comunicación con servicios web

- El certificado digital debe ser emitido por una entidad **certificadora autorizada por la superintendencia de industria y comercio**.
- El certificado digital debe acreditar al suscriptor para: **firma digital para persona Jurídica.**
- El certificado digital debe estar vigente y no estar registrado como revocado.

- Tamaño de las claves de mínimo 2048 Bits