Crypto-Vault® Digital Envelope

Fecha documento: junio 08 de 2022

Autor: Damián Salcedo

El formato Digital Envelope de Crypto-Vault® funciona firmando la información a ser protegida con una llave privada RSA, y luego procediendo a cifrar dicha información con una clave simétrica AES de 256 bits. Finalmente se cifra la clave del mensaje haciendo uso del certificado digital X.509 del receptor del mensaje (haciendo uso del algoritmo RSA).

Este sobre criptográfico fue diseñado por Milton Quiroga y es propiedad intelectual de Cybertech de Colombia Ltda (Cyte). Usando la herramienta criptográfica Crypto-Vault® podrá hacer fácilmente el proceso de protección y procesamiento de los archivos usando este y otros protocolos de protección de archivos. Para obtener más información de la herramienta y una cotización puede contactar a info@cyte.co.

Se desaconseja el tratar de implementar el protocolo criptográfico al interior de su organización si no cuenta con desarrolladores especializados en criptografía y desarrollo seguro, esto debido a que la mala implementación del mismo puede resultar en fallos de seguridad que vulnere la información del proceso de negocio que está tratando de proteger (fraudes o exposición de la información). Si adquiere la herramienta su tiempo de despliegue se hace más rápido, y la licencia de uso viene con horas de soporte para tener ingenieros criptográficos especializados a su disposición.

A continuación se explica el contenido del archivo XML que conforma un mensaje criptográfico en formato CRYPTO-VAULT DIGITAL ENVELOPE.

Estructura del sobre digital XML

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE envelope SYSTEM "envelope.dtd">

<envelope> : etiqueta raíz del sobre, que contiene todos los demás elementos.

<version>: La versión del Sobre Digital. Valor esperado “1”.

<identifier>: Identificador del sobre. Generado concatenando el serial del certificado del firmante y un número aleatorio (o pseudoaleatorio seguro).

<timestamp>: Marca de tiempo del momento en que fue generado el sobre digital.

<recipientInfo>:Información del receptor compuesto por lo siguientes tags:

<certificateInfo>: Información del certificado hacia el cual fue cifrado el sobre digital. Compuesto de los siguientes tags:

<issuer>: Información del DN (Distinguished Name) del certificado emisor siguiendo el estándar definido en el RFC4519. (cn=,o=,ou=,c=,st=,l=,street=)

<serial>: El serial del certificado.

<keyEncryptionAlgorithm>: Algoritmo con el cual fue cifrada la llave simétrica del mensaje.

<encryptedKey>:La llave simétrica con la cual fue cifrada la información del campo <encryptedContent> (información codificada en BASE 64). Usar la llave privada que corresponde al certificado de destino de este mensaje, con el algoritmo detallado en <keyEncryptionAlgorithm>.

<encryptedContentInfo>:Información del mensaje cifrado, definida en los siguientes tags:

<contentType>: Tipo de contenido. Este campo es SignedData, que quiere decir que la información cifrada en este mensaje está firmada.

<contentEncryptionAlgorithm>:Algoritmo con el cual fue cifrado el mensaje firmado que se encuentra en el tag <encryptedContent>.

<encryptedContent>: La información firmada que fue cifrada con la llave de mensaje que se encuentra en el tag <encryptedKey> (información codificada en BASE 64). Al descifrar esta información obtendrá un nuevo sobre digital XML (descrito a continuación en “Estructura mensaje firmado”) donde se encuentra la información (ejemplo: Archivo NACHA-M) junto a su firma digital.

Estructura mensaje firmado

Al descifrar el contenido que se encontraba en <encryptedContent> se obtiene un nuevo XML con la siguiente estructura:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE signedData SYSTEM "signedData.dtd">
```

<signedData>: etiqueta raíz del sobre, que contiene todos los demás elementos.

<version>: La versión del Sobre Digital. Valor esperado “1”.

<signerInfo>: Información de quién firmó el mensaje, compuesto de los siguientes tags:

<signatureAlgorithm>: El algoritmo de firma del mensaje.

<certificateInfo>:Información del certificado que corresponde a la llave privada usada para firmar el mensaje, definido en los tags <issuer> y <serial>

<issuer>: Información del DN (Distinguished Name) del certificado emisor siguiendo el estándar definido en el RFC4519. (cn=,o=,ou=,c=,st=,l=,street=)
<serial>: El serial del certificado.

<certificate>: El certificado que corresponde a la llave privada usada para firmar el mensaje en formato ASCII (Base64).

<contentInfo>: El contenido original que fue protegido. Viene comprimido en ZIP y codificado en Base64.

<encryptedDigest>: Es la firma digital de la información en claro que se encuentra en <contentInfo>. En sí es un hash criptográfico haciendo uso del algoritmo de firma definido en el tag <signatureAlgorithm>, el cual luego es firmado con la llave privada que corresponde al certificado que se encuentra en el tag <certificate>. Se debe usar dicho certificado para verificar la firma digital. La información se encuentra codificada en Base64.

PASO A PASO PARA FIRMAR Y CIFRAR:

Primer XML - Contenido firmado:

1. Tomar la información a proteger (por ejemplo, un archivo NACHA-M) y comprimirla en ZIP, luego codificar dicho ZIP en BASE64. Esta información se pone en <contentInfo>
2. Sobre la información de <contentInfo> se aplica el algoritmo de firma criptográfica que esté determinado por la llave privada que tengan configurada (ejemplo, RSAsha512). La firma que se produce queda en <encryptedDigest> codificada en base64
3. Completar la información de los demás campos xml (<signatureAlgorithm>, <issuer>, <serial> …

Segundo XML - Contenido cifrado:

4. Generar una llave aleatoria AES. Cifrar esta llave con la llave pública (RSA) del certificado del destinatario del mensaje. Esta llave cifrada se coloca codificada en base64 en <encryptedKey>
5. Generar el <identifier> concatenando el serial del certificado del firmante y un número aleatorio (o pseudoaleatorio seguro).
6. Generar el vector de inicialización (Initialization Vector o IV) usado al momento de hacer el cifrado AES del XML que contiene la firma y la información a proteger (XML de contenido firmado). El algoritmo del IV es AES/ECB/PKCS5padding. Los bytes del campo identifier se deben sacar de este instanciado como BigInteger y no como String. El IV es el resultado de cifrar con AES los 16 bytes menos significativos del identifier usando la llave AES del paso 4.
7. Utilizar la llave AES y el IV para cifrar el XML con el contenido firmado producido al final del paso 3. Este contenido cifrado se codifica en base 64 y se deja en el tag <encryptedContent>.
8. Completar la información de los demás campos xml (<keyEncryptionAlgorithm>, <issuer>, <serial> …)

PASO A PASO PARA DESCIFRAR Y VERIFICAR:

1. Obtener la llave AES del mensaje. Decodificar en base64 la llave cifrada que se encuentra en <encryptedKey>. Luego descifrar la llave con la llave privada (RSA) del certificado del destinatario del mensaje. (El campo <keyEncryptionAlgorithm> da el algoritmo a utilizar)

2. Generar el vector de inicialización (Initialization Vector o IV). El algoritmo del IV es AES/ECB/PKCS5padding. Los bytes del campo identifier se deben sacar de este instanciado como BigInteger y no como String. El IV es el resultado de cifrar con AES los 16 bytes menos significativos del identifier usando la llave AES del paso 1. Utilizar la llave AES y el IV para descifrar el XML con el contenido firmado. Este contenido cifrado se decodifica en base 64.

3. Verificar la firma digital que se encuentra en <encryptedDigest> del XML descifrado. Para ello utilizar el algoritmo de firma digital detallado en <signatureAlgorithm> y la llave pública del certificado del emisor del mensaje.

4. Decodificar el mensaje que se encuentra en <contentInfo> y luego descomprimirlo ya que viene en ZIP. Si la verificación de firma del paso 3 no fue exitosa, se debería rechazar el mensaje pues fue alterado o no proviene del emisor esperado.

Contacto: soporte@cyte.co
https://www.cyte.co/