viafirma fortress

manual de integración

Tabla de contenido

Introducción 1.1

Operaciones de consulta 1.2

Operaciones de firma 1.3

Operaciones de firma desatendida 1.4

Operaciones de extensión de firma 1.5

API 1.6

- Ping 1.6.1
- Información del usuario 1.6.2
- Certificados 1.6.3
- Firma 1.6.4
- Firma desatendida 1.6.5
- Extensión firma 1.6.6
- Encriptación / Desencriptación 1.6.7

Ejemplos rápidos de uso 1.7

Aplicación de ejemplo 1.8

Fortress Desktop (CSP) 1.9

Manual de integración de Viafirma Fortress

El presente documento pretende ser una guía de integración entre sistemas clientes y **Viafirma Fortress**.

Esta documentación se encuentra disponible en formato PDF en el siguiente enlace:
https://doc.viafirma.com/viafirma-fortress/integration/latest/es/documentation.pdf

Nota: Esta documentación técnica está sujeta a modificaciones diarias, y alguna información o configuración avanzada podría no estar reflejada.

Requisitos

Cualquier sistema cliente que quiera integrarse con Viafirma Fortress debe estar dado de alta y disponer de un `client_id` y un `client_secret`, que se utilizarán durante la integración.

Estas operaciones de gestión de sistemas clientes se realizan desde la interfaz de administración de Viafirma Fortress. Para más información al respecto, consultar el Manual de Administración de la herramienta.

Última revisión: Noviembre-2024

Autenticación del usuario y autorización de operaciones de consulta

Para que un usuario pueda interactuar con Fortress, es necesario que el mismo se autentique empleando 1 o 2 factores de autenticación (también llamado IdP: proveedor de identidad), según se determine en la configuración del sistema cliente original en Viafirma Fortress. Dependiendo de la configuración del cliente en Viafirma Fortress, fortress solicitará que el usuario se autentique contra uno o dos factores de autenticación de distinta categoría. Las categorías serán:

- Algo que sé → Knowledge
- Algo que tengo → Possession
- Algo que soy → Inherence

Solicitud de autorización

Para consumir los servicios proporcionados por Viafirma Fortress es necesario que el mismo se autentique con 1 o 2 factores de autenticación. Dependiendo de la configuración asociada al cliente de Viafirma Fortress, Viafirma Fortress puede solicitar un factor de autenticación o por el contrario Fortress forzará a que el usuario se autentique contra dos factores de autenticación de distinta categoría.

Para ello, Viafirma Fortress ofrece una interfaz web, disponible en:

```
{viafirma_fortress_url}/oauth2/v1/auth
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Una vez que se accede a esta URL, Viafirma Fortress mostrará la pantalla que permite identificar a los usuarios mediante alguno de los factores de autenticación en los que dicho usuario esté enrolado dentro de Viafirma Fortress.

Esta URL recibe una serie de parámetros, que configuran y preparan la petición de autenticación y autorización:

```
{viafirma_fortress_url}/oauth2/v1/auth?
scope=profile|certificate|certificates&
state=&
redirect_uri={url_de retorno_definido_en_viafirma_fortress}&
response_type=code&
client_id={codigo_del_cliente_definido_en_viafirma_fortress}&
user_code={codigo_del_usuario_en_viafirma_fortress}
```

Parámetro	Valor	Descripción
scope	profile / certificate / certificates	**profile**: Para servicios asociado a la información personal del usuario. En la cabecera de la pantalla aparecerá el siguiente mensaje: El sistema **CRM Acme Inc.** está solicitando su autorización para: Obtener la información de su perfil **certificate**: Para servicios asociados al acceso, permite consultar la información de un certificado del usuario seleccionado. En la cabecera de la pantalla aparecerá el siguiente mensaje: El sistema **CRM Acme Inc.** está solicitando su autorización para: Obtener la información de uno de sus certificados **certificates**: Para servicios asociados al acceso, permite consultar la información de todos los certificados del usuario seleccionado. En la

		cabecera de la pantalla aparecerá el siguiente mensaje: El sistema CRM Acme Inc. está solicitando su autorización para: Obtener la información de sus certificados
state	String	Valor opcional, permite enviar un parámetro que será devuelto a la URL de retorno del cliente sin modificación alguna por parte de Viafirma Fortress
redirect_uri	URL	Debe coincidir con una de las URL de retorno definidas en Viafirma Fortress
response_type	code	El valor debe ser **code** para las aplicaciones web cliente
client_id	Identificador del cliente	Definido en Viafirma Fortress, identifica a la aplicación cliente que está realizado la petición
user_code	Código de usuario	Indica el usuario que debe autorizar la operación

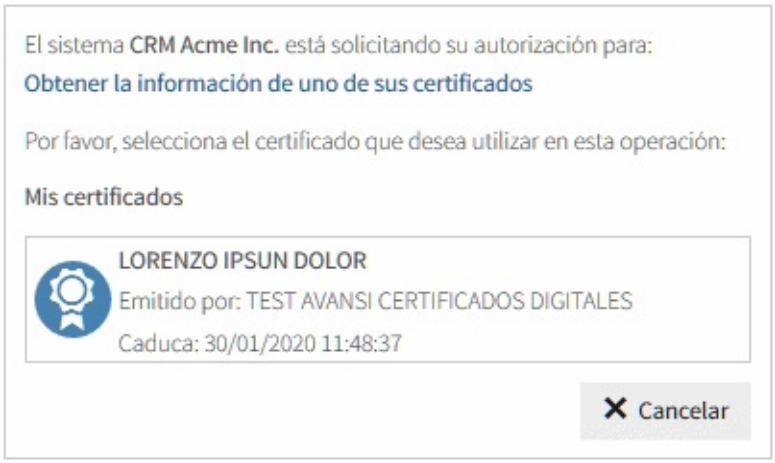
Factores de autenticación

Viafirma Fortress, mediante los diferentes factores de autenticación en los que el usuario esté enrolado, deberá asegurar la identidad del usuario.

Los factores de autenticación activos se pueden determinar durante la instalación de Viafirma Fortress, modificando los valores de los atributos correspondientes, que siguen un patrón del tipo `fortress.idp.{codigo_del_idp}.active` (ver manual de instalación).

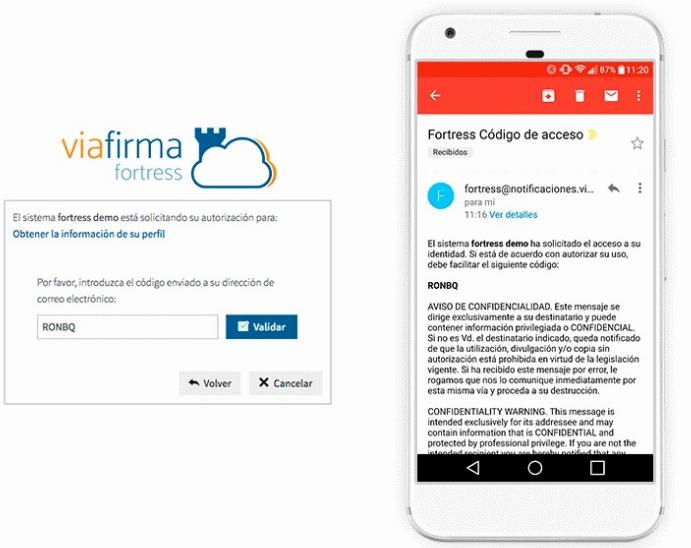
Independientemente del Proveedor de Identidad seleccionado, en caso de éxito en la autenticación, se entiende que el usuario ha autorizado la operación y se devolverá el control a la aplicación cliente, redireccionando a la URL de retorno especificada en la configuración de la petición.

Para peticiones con scope de tipo certificate, una vez que el usuario se ha autenticado correctamente usando alguno de los factores de autenticación disponibles, se mostrará la lista de certificados del usuario (custodiados por Viafirma Fortress). Una vez que el usuario ha seleccionado uno de sus certificados, se devolverá el control a la aplicación cliente.



Proveedor de Identidad: Email

Se envía al email del usuario un código único, que deberá introducir en la pantalla de autorización una vez que lo reciba.



Proveedor de Identidad: SMS

Se envía al teléfono móvil del usuario un SMS con un código único, que deberá introducir en la pantalla de autorización una vez que lo reciba.

Proveedor de Identidad: OTP

Es necesario disponer de la app (Android/IOS) que generará un código, actualizado cada cierto tiempo. El usuario deberá introducir el código en la pantalla de autorización antes de que el código expire.

Proveedor de Identidad: LDAP

Se solicitará la contraseña LDAP del usuario (la configuración del servicio LDAP se realiza durante la instalación de Viafirma Fortress).

Proveedor de Identidad: PIN

Se solicitará el código PIN que se generó y envió por e-Mail al usuario cuando se le enroló en este sistema de identificación.

Proveedor de Identidad: Password

Se solicitará la password del usuario de Viafirma Fortress.

Proveedor de Identidad: PUSH

Se solicitará un código al usuario enviado a través de una notificación PUSH a la app de su dispositivo móvil.

Obteniendo la respuesta del proceso de autenticación/autorización

Una vez que el usuario ha autorizado la operación mediante una autenticación con alguno de los factores de autenticación de Viafirma Fortress si el primer factor de autenticación ha sido delegado en el sistema cliente o contra dos factores de autenticación en Viafirma Fortress, el control vuelve al sistema cliente.

Respuestas válidas

Si el usuario **aprueba el acceso** (autenticándose correctamente usando cualquiera de los factores de autenticación), entonces la redirección de la página de autorización contiene un **código de autorización** en uno de sus parámetros (concretamente en el parámetro `code`).

```
{redirect_uri}?state=&code={codigo_de_autorizacion}
```

Ejemplo:

```
https://example.com/response?state=&code=9a3fff39-079c-45ec-b263-7d80afb18161
```

Respuestas en caso de error

Si el usuario **no aprueba el acceso** (u ocurre cualquier error durante el proceso), la redirección de la página de autorización contendrá un código de error en el parámetro `error`:

```
{redirect_uri}?error={codigo_de_error}&state=
```

Ejemplo:

```
http://example.com/?error=access_denied&state=
```

Obtención del token de acceso

Una vez obtenido el **código de autorización** es necesario canjearlo por un **token de acceso**. Para ello se realiza una petición REST (método POST) a Viafirma Fortress:

Método: `POST`
URL: `{viafirma_fortress_url}/oauth2/v1/token`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Parámetros:

Atributo	Descripción
code	Código de autorización obtenido a través de la página de autorización
client_id	Código del sistema cliente (definido en la interfaz de administración de Viafirma Fortress)
client_secret	Código secreto (client secret) del sistema cliente (definido en la interfaz de administración de Viafirma Fortress)
redirect_uri	Cualquiera de las URL de retorno definidas como válidas para el sistema cliente (definidas en la interfaz de administración de Viafirma Fortress)
grant_type	Este parámetro debe contener el valor `authorization_code` , ya que es ese tipo de token el que se está solicitando, salvo para obtener un token válido para el cliente, en cuyo caso deberemos emplear `client_credentials`

La repuesta a la petición será de tipo `application/json` , con el siguiente formato:

```
{
  "access_token": "1/fFAGRNJru1FTz70BzhT3Zg",
  "expires_in": 3920,
  "token_type": "Bearer",
  "user_code": "11111111H"
}
```

Si la solicitud de token está asociado al cliente, el resultado será

```
{
  "access_token": "1/fFAGRNJru1FTz70BzhT3Zg",
  "expires_in": 3920,
  "token_type": "Bearer"
}
```

Descripción de la respuesta:

Campo	Descripción
access_token	El token de acceso proporcionado por Viafirma Fortress
expires_in	Tiempo de vida del token de acceso (en segundos). Para aquellas peticiones de autorización que se hicieron para un número indeterminado de firmas (esto es: un `scope` de tipo `certificate` y un valor de `signatures` con el valor `"*"`), este parámetro no vendrá informado ya que el token no expira
token_type	Tipo de token retornado. El valor será siempre: `Bearer`
user_code	(token de usuario) Código del usuario en Viafirma Fortress. El sistema cliente debe comprobar que coincide con el usuario de la petición de autorización
certificate	Devuelve la información del certificado seleccionado en la solicitud de autorización si se ha indicado el valor 'certificate' en el parámetro `scope`

Acceso a las API

Una vez que se ha obtenido el token de acceso (`access_token`), la aplicación cliente puede acceder a los diferentes métodos del API que proporciona Viafirma Fortress enviando ese token en cada petición futura.

Dependiendo del `scope` seleccionado durante la petición de autorización se tendrá acceso a unas determinadas APIS:

- Para `profile` : Métodos para obtener la información del usuario
- Para `certificate` o `certificates` : Métodos para obtener la información del certificado o certificados seleccionados durante la fase de autorización.
- Para `client` : Métodos para realizar firma de documentos

Autenticación del usuario y autorización de operaciones de firma

El proceso de autenticación y autorización de operaciones de firma para un usuario, requiere la realización de los siguientes pasos:

- Autenticación del sistema Cliente.
- Solicitud de firma
- Autenticación y autorización de la solicitud
- Ejecución de la firma.

Acontinuación se describen los siguientes apartados del proceso.

Autenticación del sistema Cliente

Para realizar operaciones de firma proporcionadas por Viafirma Fortress es necesario obtener un token asociado al cliente.

Para ello, Viafirma Fortress ofrece el siguiente método Rest, disponible en:

```
{viafirma_fortress_url}/oauth2/v1/token
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Esta URL recibe una serie de parámetros, que configuran y preparan la petición de Firma realizada por un cliente:

```
{viafirma_fortress_url}/oauth2/v1/token?
scope=client&
redirect_uri={url_de retorno_definido_en_viafirma_fortress}&
client_id={codigo_del_cliente_definido_en_viafirma_fortress}&
client_secret={clave_del_cliente_definido_en_viafirma_fortress}&
grant_type=client_credentials
```

Parámetro	Valor	Descripción
scope	client	Para servicios asociado a firma de documentos.
redirect_uri	URL	Debe coincidir con una de las URL de retorno definidas en Viafirma Fortress
client_id	Client ID definido en Viafirma Fortress	Identifica a la aplicación cliente que está realizado la petición
client_secret	Clave del cliente definido en Viafirma Fortress	permite validar a la aplicación cliente que está realizado la petición
grant_type	client_credentials	Indica que el cliente solicita acceso a recursos protegidos bajo su control

Como resultado Viafirma Fortress, devolverá un objeto en formato `application/json` con la información del token de acceso asociado al cliente.

```
{
  "access_token": "1479cc2592a84cfb83c01402df613d01",
  "token_type": "Bearer",
```

```
  "expires_in": 3599
}
```

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato application/json) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Valor	Descripción
error	string	Código del error
error_description	string	Descripción del error

Posibles errores:

Código de error	Error
invalid_request	Petición incorrecta. Alguno de los parámetros de entrada no es correcto. (HTTP Status: 400)
invalid_client	Los parámetros asociados al cliente no son correctos(HTTP Status: 401)
redirect_uri_mismatch	La URL indicada en el redirect_uri no está permitida (HTTP Status: 400)
invalid_grant	Error al validar los permisos asociados al token (HTTP Status: 400)

Solicitud de firma

Con el token de sistema cliente obtenido de la anterior llamada, el cliente llamará al método /signature de Viafirma Fortress, proporcionándole la información a firmar digitalmente por parte del usuario.

En la siguiente sección encontrará la descripción en detalle del método signature, así como de los parámetros que recibe.

Una vez procesada la información Viafirma Fortress devolverá al sistema cliente un objeto en formato `application/json`, compuesto por un código de autorización y un código de ejecución

```
{
  "authCode": "124d6a9b5eaa470396a4db454780f6da",
  "exeCode": "96f1e73e5718438c8683846a2479d198"
}
```

Autenticación y autorización de la solicitud.

Una vez preparado el documento o los documentos a firmar, será necesario autenticar al usuario para poder realizar la firma.

Al igual que en el proceso a autenticación y autorización en operaciones de consulta, es necesario que el mismo se autentique con 1 o 2 factores de autenticación. Dependiendo de la configuración asociada al cliente de Viafirma Fortress, Viafirma Fortress puede solicitar un factor de autenticación o por el contrario Fortress forzará a que el usuario se autentique contra dos factores de autenticación de distinta categoría. Las categorías serán:

- Algo que sé → Knowledge

- Algo que tengo → Possession
- Algo que soy → Inherence

Para realizar el proceso de autenticación de un usuario, Viafirma Fortress ofrece una interfaz web, disponible en:

```
{viafirma_fortress_url}/oauth2/v1/auth
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Dicha URL recibe una serie de parámetros, que configuran y preparan la petición de autenticación y autorización en el proceso de firma:

```
{viafirma_fortress_url}/oauth2/v1/auth?
signature_code={codigo_autorización_de_la_firma}
scope=signature&
client_id={codigo_del_cliente_definido_en_viafirma_fortress}&
redirect_uri={url_de retorno_definido_en_viafirma_fortress}
```

Parámetro	Valor	Descripción
signature_code	Código de autorización de la firma	Código de autorización de la operación de firma
scope	signature	**signature**: Para servicios asociado a la firma de documentos
redirect_uri	URL	Debe coincidir con una de las URL de retorno definidas en Viafirma Fortress
client_id	Client ID definido en Viafirma Fortress	Identifica a la aplicación cliente que está realizado la petición

Solicitar usuario a firmar

Si el cliente no informó el campo user_code asociado al usuario, en el objeto en formato `application/json` que empleó como parámetro en la llamada del método `/signature` , Viafirma Fortress solicitará el código de usuario que desea realizar la firma.

Cuando el usuario introduzca su código de usuario en Fortress, Viafirma Fortress lo validará y le mostrará el conjunto de factores de autenticación en los que en los que el usuario se encuentra enrolado.

Viafirma Fortress almacenará el usuario una vez validado por al menos un Factor de autenticación en las cookies del navegador, para no tener que repetir el proceso cada vez que el usuario intente interactuar con Viafirma Fortress.

Factores de autenticación

Viafirma Fortress, mediante los diferentes factores de autenticación en los que el usuario esté enrolado, deberá asegurar la identidad del usuario.

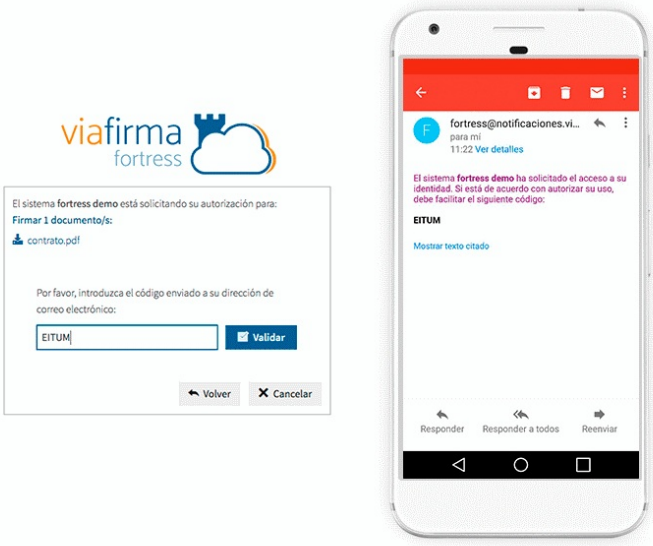
Los factores de autenticación activos se pueden determinar durante la instalación de Viafirma Fortress, modificando los valores de los atributos correspondientes, que siguen un patrón del tipo `fortress.idp.{codigo_del_idp}.active` (ver manual de instalación).

Durante todo el proceso de firma de documentos, el usuario podrá ver el número de documentos a firmar así como descargar los mismos.

Independientemente del Proveedor de Identidad seleccionado, en caso de éxito en la autenticación, se entiende que el usuario ha autorizado la operación y se devolverá el control a la aplicación cliente, redireccionando a la URL de retorno especificada en la configuración de la petición.

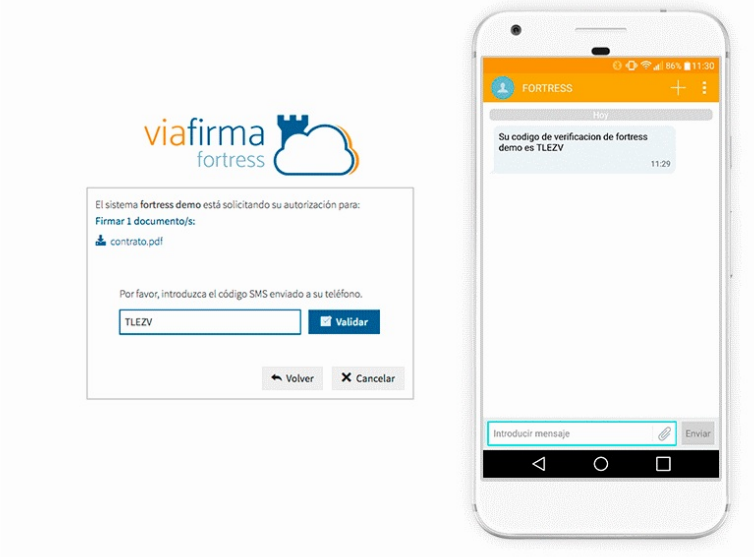
Proveedor de Identidad: Email

Se envía al email del usuario un código único, que deberá introducir en la pantalla de autorización una vez que lo reciba.



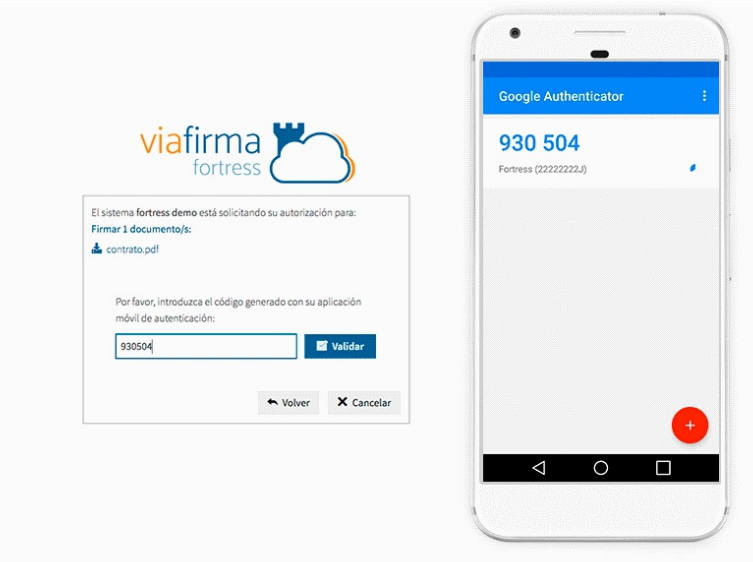
Proveedor de Identidad: SMS

Se envía al teléfono móvil del usuario un SMS con un código único, que deberá introducir en la pantalla de autorización una vez que lo reciba.



Proveedor de Identidad: OTP

Es necesario disponer de la app (Android/IOS) que generará un código, actualizado cada cierto tiempo. El usuario deberá introducir el código en la pantalla de autorización antes de que el código expire.



Proveedor de Identidad: LDAP

Se solicitará la contraseña LDAP del usuario (la configuración del servicio LDAP se realiza durante la instalación de Viafirma Fortress).

Proveedor de Identidad: PIN

Se solicitará el código Pin del usuario almacenado en Viafirma Fortress.

Proveedor de Identidad: Password

Se solicitará la password del usuario en Fortress.

Proveedor de Identidad: PUSH

Se solicitará un código al usuario enviado a través de una notificación PUSH a la app de su dispositivo móvil.

Seleccionar el certificado a emplear en la firma

Una vez que el usuario se ha autenticado correctamente usando alguno de los factores de autenticación disponibles, se mostrará la lista de certificados y certificados delegados del usuario (custodiados por Viafirma Fortress). Una vez que el usuario ha seleccionado uno de sus certificados, se devolverá el control a la aplicación cliente.

Ejecución de la firma

Finalmente cuando el usuario seleccione un certificado, Viafirma Fortress devuelve al sistema cliente la siguiente información, para que ejecute la firma:

- el certificado seleccionado
- el estado de la ejecución
- y la fecha de ejecución

Operaciones de firma desatendida

El proceso a realizar para realizar operaciones de firma desatendida, requiere la realización de los siguientes pasos:

- Autenticación del sistema Cliente.
- En el backend de Viafirma Fortress, es necesario subir el certificado que se empleará en la firma desatendida, asociado al sistema cliente o al grupo.
- Solicitud de firma.
- Ejecución de la firma.

Acontinuación se describen los siguientes apartados del proceso.

Autenticación del sistema Cliente

Para realizar operaciones de firma proporcionadas por Viafirma Fortress es necesario obtener un token asociado al cliente.

Para ello, Viafirma Fortress ofrece el siguiente método Rest, disponible en:

```
{viafirma_fortress_url}/oauth2/v1/token
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Esta URL recibe una serie de parámetros, que configuran y preparan la petición de Firma realizada por un cliente:

```
{viafirma_fortress_url}/oauth2/v1/token?
scope=client&
redirect_uri={url_de retorno_definido_en_viafirma_fortress}&
client_id={codigo_del_cliente_definido_en_viafirma_fortress}&
client_secret={clave_del_cliente_definido_en_viafirma_fortress}&
grant_type=client_credentials
```

Parámetro	Valor	Descripción
scope	client	Para servicios asociado a firma de documentos.
redirect_uri	URL	Debe coincidir con una de las URL de retorno definidas en Viafirma Fortress
client_id	Client ID definido en Viafirma Fortress	Identifica a la aplicación cliente que está realizado la petición
client_secret	Clave del cliente definido en Viafirma Fortress	permite validar a la aplicación cliente que está realizado la petición
grant_type	client_credentials	Indica que el cliente solicita acceso a recursos protegidos bajo su control

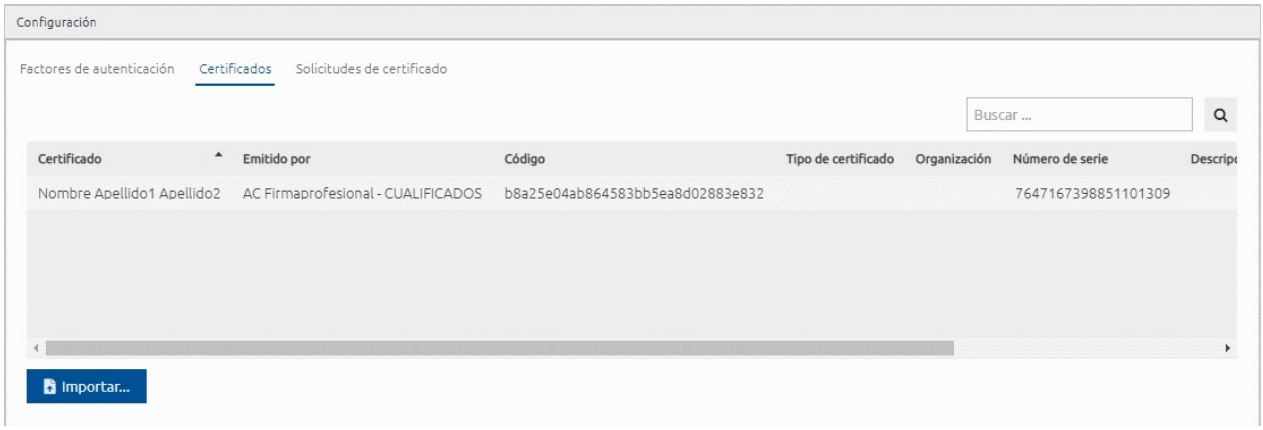
Como resultado Viafirma Fortress, devolverá un objeto en formato `application/json` con la información del token de acceso asociado al cliente.

```
{
  "access_token": "1479cc2592a84cfb83c01402df613d01",
  "token_type": "Bearer",
  "expires_in": 3599
}
```

Alojar el certificado que se empleará en el proceso, en Viafirma Fortress

Viafirma Fortress, debe gestionar los certificados que se emplearán en el proceso de firma desatendida al nivel de Sistema cliente o al nivel de Grupo. Para gestionar los certificados al nivel de cliente o grupo, será necesario:

- Acceder al backend con un usuario administrador global o de grupo
- Acceder a la administración de sus sistemas clientes o grupos
- Acceder al detalle del sistema cliente o del grupo que alojará el certificado empleado en el proceso de firma desatendida
- En la sección configuración, pulsaremos sobre la pestaña **Certificados** para consultar los certificados disponibles
- Pulsaremos importar para subir un certificado en formato `.P12`.
- Si la plataforma está configurado para solicitar certificados a una entidad de registro embebida, podrá solicitar un nuevo certificado.




```
Nota:
Es importante el valor indicado en la columna "Código", dicho valor se empleará en la solicitud de firma desatentida.
```

Solicitud de firma

Con el token de sistema cliente obtenido de la anterior llamada, el cliente llamará al método /signature de Viafirma Fortress, proporcionándole la información a firmar digitalmente de forma desatendida.

En la siguiente sección encontrará la descripción en detalle del método signature, así como de los parámetros que recibe.

Una vez procesada la información Viafirma Fortress devolverá al sistema cliente un objeto en formato `application/json`, compuesto por un código de autorización y un código de ejecución

```
{
  "authCode": "124d6a9b5eaa470396a4db454780f6da",
  "exeCode": "96f1e73e5718438c8683846a2479d198"
}
```

Ejecución de la firma

Finalmente cuando el usuario seleccione un certificado, Viafirma Fortress devuelve al sistema cliente la siguiente información, para que ejecute la firma:

- el certificado seleccionado
- el estado de la ejecución
- y la fecha de ejecución

Operaciones de extensión de firma

El proceso de operaciones de extensión de firma para un usuario, requiere la realización de los siguientes pasos:

- Autenticación del sistema Cliente.
- Realizar la solicitud de extensión de firma

Acontinuación se describen los siguientes apartados del proceso.

Autenticación del sistema Cliente

Para realizar operaciones de firma proporcionadas por Viafirma Fortress es necesario obtener un token asociado al cliente.

Para ello, Viafirma Fortress ofrece el siguiente método Rest, disponible en:

```
{viafirma_fortress_url}/oauth2/v1/token
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Esta URL recibe una serie de parámetros, que configuran y preparan la petición de Firma realizada por un cliente:

```
{viafirma_fortress_url}/oauth2/v1/token?
scope=client&
redirect_uri={url_de retorno_definido_en_viafirma_fortress}&
client_id={codigo_del_cliente_definido_en_viafirma_fortress}&
client_secret={clave_del_cliente_definido_en_viafirma_fortress}&
grant_type=client_credentials
```

Parámetro	Valor	Descripción
scope	client	Para servicios asociado a firma de documentos.
redirect_uri	URL	Debe coincidir con una de las URL de retorno definidas en Viafirma Fortress
client_id	Identificador del cliente	Se define en Viafirma Fortress e identifica a la aplicación cliente que está realizado la petición
client_secret	Clave del cliente	Permite validar a la aplicación cliente que está realizado la petición
grant_type	client_credentials	Indica que el cliente solicita acceso a recursos protegidos bajo su control

Como resultado Viafirma Fortress, devolverá un objeto en formato `application/json` con la información del token de acceso asociado al cliente.

```
{
    "access_token": "1479cc2592a84cfb83c01402df613d01",
    "token_type": "Bearer",
    "expires_in": 3599
}
```

Solicitud de extensión firma

Con el token de sistema cliente obtenido de la anterior llamada, el cliente llamará al método /extend de Viafirma Fortress, proporcionándole la información necesaria para extender la firma de un documento previamente firmado digitalmente por parte del usuario.

En la siguiente sección encontrará la descripción en detalle del método extend, así como de los parámetros que recibe.

Una vez procesada la información Viafirma Fortress devolverá al sistema cliente un objeto en formato `application/json`, compuesto por una referencia y los bytes en base 64, del documento firmado


```
{
    "ref": "d8e3d98dc20e46188fd067df28048934",
    "bytesB64": "MIMBKM8GCSqGSIb3DQEHAqCDASi/MIMBKLoCAQUxDzANBglghkgBZQMEAgEFADCC1QsGCSqGSIb3DQEHAaCC1PwEgtT4JVBERi0xLjMKJcTl8
uXrp..."
}
```

API de Viafirma Fortress

Viafirma Fortress ofrece una serie de métodos para acceder a los datos y a los certificados del usuario, pudiendo operar con ellos para realizar firmas.

Es importante recordar que para acceder a estos métodos es necesario tener un **token de acceso** (`access_token`), que se obtiene mediante la autenticación y autorización previa por parte del usuario sobre el que se desea operar, para lo que hay que seguir los pasos indicados en esta sección de la documentación.

Colecciones Postman

Si ya cuentas con credenciales de acceso a nuestro entorno Sanbdox podrás ayudarte de los siguientes recursos de postman para probar el API. En estas colecciones se incluyen los casos de uso básicos con los que podrás comenzar tu integración.

En las propias colecciones de postman incluimos documentación y explicación de cada caso de uso, revisa la sección de documentación.

- Entorno de configuración Sandbox

Operaciones de Firma

- Colección Postman Fortress Signature API

Operaciones de autenticación

- Colección Postman Fortress User Authentication API

API: Métodos para validar la comunicación con el sistema

Validar la comunicación con el sistema

Con este método podemos validar la comunicación con la instancia de Viafirma Fortress.

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/ping`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Ejemplo:

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/ping`

Respuesta del servicio

La respuesta de este servicio devolverá un `200 OK` si hay comunicación con la instancia de Viafirma Fortress.

Errores del servicio

Si no hay conexión con la instancia de Viafirma Fortress, se producirá un error de comunicación.

Posibles errores:

Código de error	Error
not_found	Si hay comunicación con la instancia, pero dicha versión de Viafirma Fortress no tiene implementado este método. (HTTP Status: 404)

API: Métodos relacionados con la información del usuario

Importante: Para acceder a estos métodos es necesario tener un **token de acceso** (`access_token`) obtenido mediante una petición de autenticación y autorización con un `scope` de tipo `profile` , para lo que hay que seguir los pasos indicados en esta sección de la documentación.

Obtener datos del usuario

Con este método podemos ver los datos de un usuario de Viafirma Fortress, tales como nombre, correo electrónico, teléfono móvil, sus certificados, ...

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/user/{user_code}`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `user_code` : Código del usuario del que se desea obtener la información

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `GET`
URL: `https://fortress.viafirma.com/fortress/api/v1/user/sample_user`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:


```
{
  "code": "sample_user",
  "name": "Name Surname",
  "email": "user_mail@example.com",
  "mobile": "+34666666666",
  "lastAccess": 1501590523833,
  "role": "ROLE_USER",
  "certificates": [
    {
      "code": "226ffa94-1f0f-4c43-98aa-c7c8e4ccf657",
      "name": "Sample Certificate 01",
      "description": "Lorem ipsum dolor sit amet",
      "dateIssued": 1492432671000,
      "dateExpired": 1555504674000,
      "serialNumber": "1250978750360690486",
      "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TEST AVANSI CERTIFICADOS DIGITALES",
      "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=Sample Certificate 01, SERIALNUMBER=TEST, GI
VENNAME=TEST, SURNAME=TEST, C=DO"
    },
```

```
    {
      "code": "014e684e-4751-4850-853c-c90802385a78",
      "name": "Sample Certificate 02",
      "description": "Lorem ipsum dolor sit amet",
      "dateIssued": 1492517893000,
      "dateExpired": 1555504678000,
      "serialNumber": "4096319273351924161",
      "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TEST AVANSI CERTIFICADOS DIGITALES",
      "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=Sample Certificate 02, SERIALNUMBER=TEST, GI
VENNAME=TEST, SURNAME=TEST, C=DO"
    }
  ]
}
```


Donde:

Parámetro	Tipo	Descripción
code	*string*	Código del usuario
name	*string*	Nombre completo del usuario
email	*string*	Correo electrónico del usuario
mobile	*string*	Teléfono móvil de usuario
lastAccess	*long*	Fecha de último acceso del usuario
role	*string*	Rol del usuario
certificates	*array*	Certificados del firmante

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:


```
{
  "error": "error_code",
  "error_description": "error_description"
}
```


Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
user_not_found	El usuario no es correcto o no está activo (HTTP Status: 404)

Obtener estado del usuario

Con este método podemos ver el estado de un usuario de Viafirma Fortress. Este estado nos indicará, entre otras cosas, si el usuario está activo, si dispone de algún certificado, si está enrolado en algún Proveedor de Identificación…

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/user/{user_code}/status`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `user_code` : Código del usuario del que se desea obtener el estado

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `GET`
URL: `https://fortress.viafirma.com/fortress/api/v1/user/sample_user/status`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
{
  "sign": true,
  "auth": true
}
```

Donde:

Parámetro	Tipo	Descripción
sign	*boolean*	Indica si el usuario puede realizar operaciones de firma
auth	*boolean*	Indica si el usuario puede realizar operaciones de autenticación

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
user_not_found	El usuario no es correcto o no está activo (HTTP Status: 404)

Eliminar la caché de credenciales

Con este método podemos eliminar la caché de credenciales de un usuario de Viafirma Fortress.

Uso del servicio

Método: `DELETE`
URL: `{viafirma_fortress_url}/api/v1/user/{user_code}/removeAuthCache`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `user_code` : Código del usuario del que se desea obtener la información

Además, en la cabecera HTTP de la petición `DELETE` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método `DELETE`
URL: `https://fortress.viafirma.com/fortress/api/v1/user/sample_user/removeAuthCache`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
user_not_found	El usuario no es correcto o no está activo (HTTP Status: 404)

API: Métodos relacionados con los certificados del usuario

Importante: Para acceder a estos métodos es necesario tener un **token de acceso** (`access_token`) obtenido mediante una petición de autenticación y autorización con un `scope` de tipo `certificate` o `certificates` , para lo que hay que seguir los pasos indicados en esta sección de la documentación.

Obtener certificados del usuario

Con este método podemos obtener el listado de certificados disponibles (activos) para un usuario.

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/user/{user_code}/certificate`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `user_code` : Código del usuario del que se desea obtener los certificados

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `GET`
URL: `https://fortress.viafirma.com/fortress/api/v1/user/sample_user/certificate`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
[
    {
        "code": "226ffa94-1f0f-4c43-98aa-c7c8e4ccf657",
        "name": "Sample Certificate 01",
        "description": "Lorem ipsum dolor sit amet"
        "dateIssued": 1492432671000,
        "dateExpired": 1555504674000,
        "serialNumber": "1250978750360690486",
        "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TEST AVANSI CERTIFICADOS DIGITALES",
        "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=Sample Certificate 01, SERIALNUMBER=TEST,
GIVENNAME=TEST, SURNAME=TEST, C=DO",
        "pem": "MIIGsTCCBZmgAwIBAgIQESeGCdXLzw9XurB4LNd0BjANBgkq..."
    },
    {
        "code": "014e684e-4751-4850-853c-c90802385a78",
        "name": "Sample Certificate 02",
        "description": "Lorem ipsum dolor sit amet"
        "dateIssued": 1492517893000,
        "dateExpired": 1555504678000,
        "serialNumber": "4096319273351924161",
```

```
        "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TEST AVANSI CERTIFICADOS DIGITALES",
        "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=Sample Certificate 02, SERIALNUMBER=TEST,
GIVENNAME=TEST, SURNAME=TEST, C=DO",
        "pem": "MIIFTDCCBDSgAwIBAgIIHZer06chPs4wDQYJKoZIhvcNAQEFB..."
    },
    {
        "code": "024v694e-4899-4876-863f-j91872310e70",
        "name": "Sample Certificate 03",
        "description": "Lorem ipsum dolor sit amet"
        "dateIssued": 1493432678000,
        "dateExpired": 1556504679000,
        "serialNumber": "2046339272352914110",
        "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TEST AVANSI CERTIFICADOS DIGITALES",
        "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=Sample Certificate 03, SERIALNUMBER=TEST,
GIVENNAME=TEST, SURNAME=TEST, C=DO",
        "pem": "MIIGnTCCBYWgAwIBAgIQTuF2zDNK0C5XVqAhuNMuHjANBgkqhk..."
    }
]
```


Donde:

Parámetro	Tipo	Descripción
code	*string*	Código identificador del certificado
name	*string*	Nombre del certificado
description	*string*	Descripción del certificado
dateIssued	*string*	Fecha de expedición del certificado en milisegundos
dateExpired	*string*	Fecha de expiración del certificado en milisegundos
serialNumber	*string*	Número de serie asociado al certificado
issuer	*string*	Emisor del certificado
subject	*string*	Sujeto del certificado
pem	*string*	Certificado (público) en formato PEM

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
user_not_found	El usuario no es correcto o no está activo (HTTP Status: 404)

Obtener un certificado del usuario

Con este método podemos obtener un certificado en concreto disponible (activo) para un usuario.

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/user/{user_code}/certificate/{certificate_code}`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `user_code` : Código del usuario del que se desea obtener el certificado
- `certificate_code` : Código del certificado del que se desea obtener la información

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `GET`
URL: `https://fortress.viafirma.com/fortress/api/v1/user/sample_user/certificate/226ffa94-1f0f-4c43-98aa-c7c8e4ccf657`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
[
    {
        "code": "226ffa94-1f0f-4c43-98aa-c7c8e4ccf657",
        "name": "Sample Certificate 01",
        "description": "Lorem ipsum dolor sit amet"
        "dateIssued": 1492432671000,
        "dateExpired": 1555504674000,
        "serialNumber": "1250978750360690486",
        "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TEST AVANSI CERTIFICADOS DIGITALES",
        "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=Sample Certificate 01, SERIALNUMBER=TEST,
GIVENNAME=TEST, SURNAME=TEST, C=DO",
        "pem": "MIIGsTCCBZmgAwIBAgIQESeGCdXLzw9XurB4LNd0BjANBgkq..."
    }
]
```

Donde:

Parámetro	Tipo	Descripción
code	*string*	Código identificador del certificado
name	*string*	Nombre del certificado
description	*string*	Descripción del certificado
dateIssued	*string*	Fecha de expedición del certificado en milisegundos
dateExpired	*string*	Fecha de expiración del certificado en milisegundos
serialNumber	*string*	Número de serie asociado al certificado
issuer	*string*	Emisor del certificado

subject	*string*	Sujeto del certificado
pem	*string*	Certificado (público) en formato PEM

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
user_not_found	El usuario no es correcto o no está activo (HTTP Status: 404)
certificate_not_found	El certificado no es correcto o no está activo (HTTP Status: 404)

Obtener certificados de un sistema cliente

Con este método podemos obtener el listado de certificados disponibles (activos) para un sistema cliente.

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/client/{client_id}/certificate`

Donde:

- `viafirma_fortress_url`: URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `client_id`: Identificador del sistema cliente registrado en Viafirma Fortress

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `GET`
URL: `https://fortress.viafirma.com/fortress/api/v1/client/sample_client/certificate`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:


```
[{
        "code": "08d87ff2ed124a8bb7b323cbfb889e9e",
        "dateIssued": 1555495728000,
        "dateExpired": 1618567728000,
        "serialNumber": "228897951488527728794",
        "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TEST AVANSI CERTIFICADOS DIGITALES ",
        "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=LUCASMORA PRIETO,SERIALNUMBER = 94967442 M
,GIVENNAME = LUCAS,SURNAME = MORA PRIETO,C = DO ",
        "issuerMap": {
            "C": "DO",
            "CN": "TEST AVANSI CERTIFICADOS DIGITALES",
            "L": "WWW.AVANSI.COM.DO",
            "O": "AVANSI S.R.L. - RNC 130222509"
        },
        "subjectMap": {
            "SURNAME": "MORA PRIETO",
            "C": "DO",
            "SERIALNUMBER": "94967442M",
            "1.3.6.1.4.1.27395.8.1": "CERTIFICADO DE PERSONA INDIVIDUAL",
            "CN": "LUCAS MORA PRIETO",
            "GIVENNAME": "LUCAS"
        },
        "pem": "MIIFWjCCBEKgAwI...",
        "delegated": false,
        "level": "MEDIUM"
    },
    {
        "code": "0566a9abed054800afcfc0c7e927bd40",
        "dateIssued": 1555495668000,
        "dateExpired": 1618567668000,
        "serialNumber": "2262470652122451427458",
        "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TEST AVANSI CERTIFICADOS DIGITALES ",
        "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=MARC SERRA CRESPO, SERIALNUMBER = Y1547327
Q, GIVENNAME = MARC, SURNAME = SERRA CRESPO, C = DO ",
        "issuerMap": {
            "C": "DO",
            "CN": "TEST AVANSI CERTIFICADOS DIGITALES",
            "L": "WWW.AVANSI.COM.DO",
            "O": "AVANSI S.R.L. - RNC 130222509"
        },
        "subjectMap": {
            "SURNAME": "SERRA CRESPO",
            "C": "DO",
            "SERIALNUMBER": "Y1547327Q",
            "1.3.6.1.4.1.27395.8.1": "CERTIFICADO DE PERSONA INDIVIDUAL",
            "CN": "MARC SERRA CRESPO",
            "GIVENNAME": "MARC"
        },
        "pem": "MIIFWzCCBEOgAwIBAgI...",
        "delegated": false,
        "level": "MEDIUM"
    }
]
```


Donde:

Parámetro	Tipo	Descripción
code	*string*	Código identificador del certificado
name	*string*	Nombre del certificado
description	*string*	Descripción del certificado
dateIssued	*string*	Fecha de expedición del certificado en milisegundos
dateExpired	*string*	Fecha de expiración del certificado en milisegundos

serialNumber	*string*	Número de serie asociado al certificado
issuer	*string*	Emisor del certificado
subject	*string*	Sujeto del certificado
issuerMap	*object*	Atributos del emisor
subjectMap	*object*	Atributos del sujeto
delegated	*boolean*	indicador de certificado delegado
pem	*string*	Certificado (público) en formato PEM
level	*string*	Nivel de protección del certificado

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
client_not_found	El cliente no es correcto o no está activo (HTTP Status: 404)

Obtener un certificado del cliente

Con este método podemos obtener un certificado en concreto disponible (activo) para un cliente.

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/client/{client_id}/certificate/{certificate_code}

Donde:

- `viafirma_fortress_url`: URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `client_id`: Identificador del sistema cliente registrado en Viafirma Fortress
- `certificate_code`: Código del certificado del que se desea obtener la información

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `GET`
URL: `https://fortress.viafirma.com/fortress/api/v1/client/sample_client/certificate/226ffa94-1f0f-4c43-98aa-c7c8e4ccf657`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:


```
{
  "code": "08d87ff2ed124a8bb7b323cbfb889e9e",
  "dateIssued": 1555495728000,
  "dateExpired": 1618567728000,
  "serialNumber": "228897951488527728794",
  "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TESTAVANSI CERTIFICADOS DIGITALES ",
  "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=LUCAS MORA PRIETO, SERIALNUMBER = 94967442 M, GI
VENNAME = LUCAS, SURNAME = MORA PRIETO, C = DO ",
  "issuerMap": {
    "C": "DO",
    "CN": "TEST AVANSI CERTIFICADOS DIGITALES",
    "L": "WWW.AVANSI.COM.DO",
    "O": "AVANSI S.R.L. - RNC 130222509"
  },
  "subjectMap": {
    "SURNAME": "MORA PRIETO",
    "C": "DO",
    "SERIALNUMBER": "94967442M",
    "1.3.6.1.4.1.27395.8.1": "CERTIFICADO DE PERSONA INDIVIDUAL",
    "CN": "LUCAS MORA PRIETO",
    "GIVENNAME": "LUCAS"
  },
  "pem": "MIIFWjCCBEKgAwIBAgI...",
  "delegated": false,
  "level": "MEDIUM"
}
```


Donde:

Parámetro	Tipo	Descripción
code	*string*	Código identificador del certificado
name	*string*	Nombre del certificado
description	*string*	Descripción del certificado
dateIssued	*string*	Fecha de expedición del certificado en milisegundos
dateExpired	*string*	Fecha de expiración del certificado en milisegundos
serialNumber	*string*	Número de serie asociado al certificado
issuer	*string*	Emisor del certificado
subject	*string*	Sujeto del certificado
issuerMap	*object*	Atributos del emisor
subjectMap	*object*	Atributos del sujeto
delegated	*boolean*	indicador de certificado delegado
pem	*string*	Certificado (público) en formato PEM
level	*string*	Nivel de protección del certificado

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
client_not_found	El cliente no es correcto o no está activo (HTTP Status: 404)
certificate_not_found	El certificado no es correcto o no está activo (HTTP Status: 404)

Alta de nuevos certificados de cliente

Este servicio permite registrar un nuevo certificado y asociarlo a un sistema cliente.

Uso del servicio

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/client/{client_id}/certificate

Donde:

- `viafirma_fortress_url`: URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `client_id`: Identificador del sistema cliente registrado en Viafirma Fortress

Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Ejemplo:

Método: `POST`
URL: `https://fortress.viafirma.com/fortress/api/v1/client/sample_client/certificate`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Parámetros del servicio

Este servicio recibe por parámetros la configuración del certificado a firmar:

Los parámetros que se reciben (en formato `application/json`) tienen la siguiente forma:

```
{
  "keystore": "MIIZXwIBAzCCGRgGCSq...",
  "password": "123456"
}
```

Donde:

Parámetro	Tipo	Descripción
code	*string*	[OPCIONAL] Código a asociar el certificado, si no se informa Fortress genera uno
description	*string*	[OPCIONAL] Descripción asociada al certificado
keystore	*string*	Contenido del keystore en formato PKCS#12 codificado en Base64
password	*string*	Contraseña del keystore
alias	*string*	[OPCIONAL] Alias del certificado dentro del keystore, solo es obligatorio si el keystore almacena más de un certificado

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) con los datos del certificado en el mismo formato que el servicio de consulta de un certificado de un sistema cliente.


```
{
  "code": "08d87ff2ed124a8bb7b323cbfb889e9e",
  "dateIssued": 1555495728000,
  "dateExpired": 1618567728000,
  "serialNumber": "228897951488527728794",
  "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TESTAVANSI CERTIFICADOS DIGITALES ",
  "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=LUCAS MORA PRIETO, SERIALNUMBER = 94967442 M, GI
VENNAME = LUCAS, SURNAME = MORA PRIETO, C = DO ",
  "issuerMap": {
    "C": "DO",
    "CN": "TEST AVANSI CERTIFICADOS DIGITALES",
    "L": "WWW.AVANSI.COM.DO",
    "O": "AVANSI S.R.L. - RNC 130222509"
  },
  "subjectMap": {
    "SURNAME": "MORA PRIETO",
    "C": "DO",
    "SERIALNUMBER": "94967442M",
    "1.3.6.1.4.1.27395.8.1": "CERTIFICADO DE PERSONA INDIVIDUAL",
    "CN": "LUCAS MORA PRIETO",
    "GIVENNAME": "LUCAS"
  },
  "pem": "MIIFWjCCBEKgAwIBAgI...",
  "delegated": false,
  "level": "MEDIUM"
}
```


Donde:

Parámetro	Tipo	Descripción
code	*string*	Código identificador del certificado
name	*string*	Nombre del certificado
description	*string*	Descripción del certificado
dateIssued	*string*	Fecha de expedición del certificado en milisegundos
dateExpired	*string*	Fecha de expiración del certificado en milisegundos
serialNumber	*string*	Número de serie asociado al certificado
issuer	*string*	Emisor del certificado
subject	*string*	Sujeto del certificado
issuerMap	*object*	Atributos del emisor
subjectMap	*object*	Atributos del sujeto

delegated	*boolean*	indicador de certificado delegado
pem	*string*	Certificado (público) en formato PEM
level	*string*	Nivel de protección del certificado

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
client_not_found	El cliente no es correcto o no está activo (HTTP Status: 404)
invalid_keystore	El keystore no se encuentra en formato PKCS#12 o la password es incorrecta (HTTP Status: 404)
invalid_alias	No se ha encontrado el certificado con el alias especificado dentro del keystore, o existen varios certificados y no se ha especificado el alias (HTTP Status: 404)
certificate_already_exists	El certificado ya se encuentra asociado al sistema cliente (HTTP Status: 404)
expired_certificate	El certificado ha caducado (HTTP Status: 404)
revoked_certificate	El certificado se encuentra revocado (HTTP Status: 404)
not_trusted_certificate	Alguno de los certificados de la cadena no se encuentra en el almacén de confianza (HTTP Status: 404)
certificate_validation	Se ha producido un error al validar el certificado (HTTP Status: 404)

Eliminación de certificados de cliente

Este servicio permite eliminar certificados asociados a un sistema cliente.

Uso del servicio

Método: `DELETE`
URL: `{viafirma_fortress_url}/api/v1/client/{client_id}/certificate/{certificate_code}

Donde:

- `viafirma_fortress_url`: URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `client_id`: Identificador del sistema cliente registrado en Viafirma Fortress
- `certificate_code`: código del certificado

Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Ejemplo:

Método `DELETE`
URL: `https://fortress.viafirma.com/fortress/api/v1/client/sample_client/certificate/08d87ff2ed124a8bb7b323cbfb889e9e` Header de la petición: Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) con los datos del certificado eliminado en el mismo formato que el servicio de consulta de un certificado de un sistema cliente.


```
{
  "code": "08d87ff2ed124a8bb7b323cbfb889e9e",
  "dateIssued": 1555495728000,
  "dateExpired": 1618567728000,
  "serialNumber": "228897951488527728794",
  "issuer": "C=DO, L=WWW.AVANSI.COM.DO, O=AVANSI S.R.L. - RNC 130222509, CN=TESTAVANSI CERTIFICADOS DIGITALES ",
  "subject": "OID.1.3.6.1.4.1.27395.8.1=CERTIFICADO DE PERSONA INDIVIDUAL, CN=LUCAS MORA PRIETO, SERIALNUMBER = 94967442 M, GI
VENNAME = LUCAS, SURNAME = MORA PRIETO, C = DO ",
  "issuerMap": {
    "C": "DO",
    "CN": "TEST AVANSI CERTIFICADOS DIGITALES",
    "L": "WWW.AVANSI.COM.DO",
    "O": "AVANSI S.R.L. - RNC 130222509"
  },
  "subjectMap": {
    "SURNAME": "MORA PRIETO",
    "C": "DO",
    "SERIALNUMBER": "94967442M",
    "1.3.6.1.4.1.27395.8.1": "CERTIFICADO DE PERSONA INDIVIDUAL",
    "CN": "LUCAS MORA PRIETO",
    "GIVENNAME": "LUCAS"
  },
  "pem": "MIIFWjCCBEKgAwIBAgI...",
  "delegated": false,
  "level": "MEDIUM"
}
```


Donde:

Parámetro	Tipo	Descripción
code	*string*	Código identificador del certificado
name	*string*	Nombre del certificado
description	*string*	Descripción del certificado
dateIssued	*string*	Fecha de expedición del certificado en milisegundos
dateExpired	*string*	Fecha de expiración del certificado en milisegundos
serialNumber	*string*	Número de serie asociado al certificado
issuer	*string*	Emisor del certificado
subject	*string*	Sujeto del certificado
issuerMap	*object*	Atributos del emisor
subjectMap	*object*	Atributos del sujeto
delegated	*boolean*	indicador de certificado delegado
pem	*string*	Certificado (público) en formato PEM

level	*string*	Nivel de protección del certificado

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Posibles errores:

Código de error	Error
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
client_not_found	El cliente no es correcto o no está activo (HTTP Status: 404)
certificate_not_found	No existe un certificado con el código especificado (HTTP Status: 404)

API: Métodos relacionados con la firma con certificado

El proceso de firma en Viafirma Fortress, constará de los procesos de:

- Autenticación del cliente
- Solicitud de firma
- Autenticación y autorización de la solicitud de firma
- Ejecución de la firma
- Obtención del documento/s firmado/s

En los siguientes apartados describiremos los métodos disponibles en Viafirma Fortress, asociados a las operaciones de firma:

Nota: Para acceder a estos métodos es necesario tener un **token de acceso** (`access_token`) obtenido mediante una petición de autenticación y autorización con un `scope` de tipo `client` y un `grant_type` de tipo `client_credentials` , para lo que hay que seguir los pasos indicados en esta sección de la documentación.

Solicitud de firma

Uso del servicio

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Además, en la cabecera HTTP de la petición `POST` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Parámetros del servicio

Este servicio recibe por parámetros la configuración de la firma empleada por cada documento a firmar, donde se indica, entre otras cosas, el tipo de firma que se quiere realizar, el documento a firmar…

Los parámetros que se reciben (en formato `application/json`) tienen la siguiente forma:

```
{
  "signatureConfigurations": [
    {
      "ref": "#1",
      "document": {
        "bytesB64": "JVBERi0xLjMKJcTl8uXrp/Og0MTGCjQ...",
        "name": "contrato.pdf"
      },
      "signatureType": "PADES_B",
      "signatureAlgorithm": "RSA_SHA256",
```

```
      "packaging": "ENVELOPED",
      "reason": " test PAdES signature ",
      "padesConfiguration": {
        "stamper": {
          "csvPath": "http://localhost:7080/fortress/v#",
          "logoB64": "iVBORw0KGgoAAAANSUhEUgAAAWYAAABsCAYAAABZyhj...",
          "page": 1,
          "type": "QR_BARCODE128",
          "xAxis": 80,
          "yAxis": 700
        }
      }
    }
  ]
}
```

Nota: Los detalles de los parámetros `padesConfiguration`, `xadesConfiguration`, `tsa` y `policy` se muestran más adelante.

Donde:

Parámetro	Tipo	Descripción
userCode	*string*	Usuario que debe de realizar la firma, si el sistema cliente no informa dicho valor, Viafirma Fortress solicitará el usuario a emplear en el proceso de autenticación y autorización de la solicitud de firma
certificateCode	*string*	Código de certificado a emplear en la firma, si el sistema cliente no informa dicho valor, Viafirma Fortress solicitará el certificado a emplear en el proceso de firma tras de autenticar al usuario
certificatePassword	*string*	Contiene la password del certificado, este campo solamente está permitido para la firma desatendida.
multifactorAuth	*boolean*	Si se indica true, fuerza al empleo de 2 factores de autenticación.
async	*string*	Si se establece con el valor `true` la llamada al servicio de ejecución de firma se realiza de forma asíncrona.
callbackUrl	*string*	Si se informa una URL y la firma se realiza de forma asíncrona, al finalizar la firma Fortress realiza una petición POST a dicha URL con el estado final de ejecución.
certificateFilter	*string*	Atributos por el que filtrar el certificado. Introduces que filtros quieres que cumpla el certificado para ser usado.
signatureConfigurations/document/name	*string*	Nombre del documento a firmar
signatureConfigurations/document/bytesB64	*string*	Documento a firmar, codificado en Base64
signatureConfigurations/document/url	*string*	URL del documento a firmar
signatureConfigurations/signatureType	*string*	Tipo de firma. Valores disponibles: - `CADES_B` - `CADES_T` - `CADES_LT` - `CADES_LTA` - `PADES_B` - `PADES_T` - `PADES_LT` - `PADES_LTA` - `XADES_B` - `XADES_T` - `XADES_LT`

		- `XADES_LTA` - `PKCS1`
signatureConfigurations/signatureAlgorithm	*string*	Algoritmo que se usará para cifrar la firma. Valores disponibles: - `RSA_SHA1` - `RSA_SHA224` - `RSA_SHA256` - `RSA_SHA384` - `RSA_SHA512`
signatureConfigurations/packaging	*string*	Envoltura de la firma. Valores disponibles: - `ENVELOPED` - `ENVELOPING` - `DETACHED`
signatureConfigurations/reason	*string*	OPCIONAL, motivo de la firma
signatureConfigurations/location	*string*	OPCIONAL, localización de la firma
signatureConfigurations/ref	*string*	Si se informa se devolverá el mismo valor en el resultado de la firma.

Configuración de los filtros de certificados

Esta configuración hace que a la hora de firmar, el usuario solo pueda firmar con los certificados que cumplan todos los requisitos.

```
{
  "certificateFilter": {
    "issuer.cn": [
      "AC FNMT Usuarios"
    ],
    "subject.cn": [
      "ZAMORANO DE EJEMPLO JOSE LUIS - 71121212M"
    ],
    "subject.serialnumber": [
        "99999999R",
        "71121212M"
    ],
    "serialnumber": [
         "7595d18d1feaad86ca2bde0f5c022b687ad74075"
    ],
    "oid": [
      "2.5.29.14",
      "2.5.29.15"
    ]
  }
}
```

Podemos filtrar de varias formas.

Parámetro	Tipo	Descripción
oid	*List - String*	Lista de OID que necesita tener el certificado para ser válido
serialnumber	*List - String*	Lista de serialnumber que necesita tener el certificado para ser válido
issuer.C	*Single list - String*	(Solo se permite uno, a partir de la versión 6.2.5 de Viafirma Fortress, se elimina esta restricción) CountryName -> ES
issuer.OU	*Single list - String*	(Solo se permite uno, a partir de la versión 6.2.5 de Viafirma Fortress, se elimina esta restricción) OrganizationalUnit -> Ceres
	Single	

issuer.CN	*list - String*	eliminasesta restricción) CommonName -> AC FNMT Usuarios
issuer.O	*Single list - String*	(Solo se permite uno, a partir de la versión 6.2.5 de Viafirma Fortress, se elimina esta restricción) Organization -> FNMT-RCM
subject.SURNAME	*Single list - String*	(Solo se permite uno, a partir de la versión 6.2.5 de Viafirma Fortress, se elimina esta restricción) APELLIDOS -> ZAMORANO DE EJEMPLO
subject.C	*Single list - String*	(Solo se permite uno, a partir de la versión 6.2.5 de Viafirma Fortress, se elimina esta restricción) CountryName -> ES
subject.SERIALNUMBER	*Single list - String*	(Solo se permite uno, a partir de la versión 6.2.5 de Viafirma Fortress, se elimina esta restricción) Numero de serie -> 71121212M
subject.CN	*Single list - String*	(Solo se permite uno, a partir de la versión 6.2.5 de Viafirma Fortress, se elimina esta restricción) CommonName -> ZAMORANO DE EJEMPLO JOSE LUIS - 71121212M
subject.GIVENNAME	*Single list - String*	(Solo se permite uno, a partir de la versión 6.2.5 de Viafirma Fortress, se elimina esta restriccióno) Nombre -> JOSE LUIS

Configuración PAdES

Esta configuración solo se aplica para las firmas cuyo `signatureType` es de tipo PAdES (PAdES B, PAdES T, PAdES LT, PAdES LTA).

```
"padesConfiguration": {
    "stamper": { }
}
```

El objeto stamper es opcional, y sirve para definir un sello visual asociado a la firma PAdES.

```
{
  "stamper": {
    "csvPath": "https://fortress.viafirma.com/fortress/v#",
    "logoB64": "JVBERi0xLjMKJcTl8uXlRU9GC...",
    "page": 1,
    "rotation": "ROTATE_90",
    "textLine1": "Sample line 1",
    "textLine2": "Sample line 2",
    "textLine3": "Sample line 3",
    "type": "QR_BARCODE128",
    "xAxis": 100,
    "yAxis": 100,
    "timeZoneId": "America/Santo_Domingo"
  }
}
```

Parámetro	Tipo	Descripción
stamper/csvPath	*string*	URL de verificación de la firma
stamper/xAxis	*int*	Posición (en el eje X) del sello de firma
stamper/yAxis	*int*	Posición (en el eje Y) del sello de firma
stamper/width	*int*	Opcional. Ancho del sello de firma
stamper/height	*int*	Opcional. Alto del sello de firma
stamper/imageB64	*string*	Imagen de fondo del sello de firma codificado en Base64

stamper/imageUrl	*string*	URL de la imagen de fondo del sello de firma
stamper/logoB64	*string*	Opcional. Logo del sello de firma (se pintará en la parte inferior derecha del sello)
stamper/page	*int*	Página donde se pintará el sello. Se puede usar el valor `-1` para indicar la última página o el valor `0` para pintar el sello en todas las páginas del documento
stamper/rotation	*string*	Rotación del sello. Si se informa, el sello rotará los grados indicados. Valores posibles: - `ROTATE_90` - `ROTATE_270`
stamper/textLine1	*string*	Primera linea textual a pintar en el contenido del sello de firma
stamper/textLine2	*string*	Segunda linea textual a pintar en el contenido del sello de firma
stamper/textLine3	*string*	Tercera linea textual a pintar en el contenido del sello de firma
stamper/type	*string*	Tipo de sello. Valores disponibles: - `PDF417` - `QR_BARCODE128` - `QR` - `BARCODE128` - `IMAGE` - `TEXT` - `QR_NO_TEXT` - `QR_SCALED` - `CUSTOM_TEXT` - `QR_REDUCED` - `CSV` - `CSV_QR` - `IMAGE_TEXT` - `DEFAULT`
stamper/timeZoneId	*string*	String que se corresponderá con la lista estándar de zonas horarias.

Si en la llamada a la API NO viene informado el valor `timeZoneId` aplicaremos el siguiente criterio:

- Si la petición pertenece a un usuario, emplearemos el timezone del usuario, si no lo tiene informado, pero pertenece a algún grupo que lo tenga informado, aplicamos el del primer grupo que lo tenga informado, si no aplicamos el configurado por defecto en el sistema.

Para el tipo de sello `IMAGE` se debe especificar la imagen de fondo del sello en formato Base64 usando el atributo `imageB64` o indicando una URL en el atributo `imageUrl` .

Configuración XAdES

Esta configuración solo se aplica para las firmas cuyo `signatureType` es de tipo XAdES (XAdES B, XAdES T, XAdES LT, XAdES LTA)

```
{
  "signedInfoCanonicalizationMethod": "http://www.w3.org/TR/2001/REC-xml-c14n-20010315",
  "signedPropertiesCanonicalizationMethod": "http://www.w3.org/TR/2001/REC-xml-c14n-20010315",
  "xPathLocationString": "//book[@id='bk101-1']",
  "claimedSignerRoles": [
    "role1",
    "role2"
  ],
  "transformAlgorithms": [
    "http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
  ],
  "dssReferenceUri": "http://dsa-reference.example.com/"
}
```

Donde:

Parámetro	Tipo	Descripción
signedInfoCanonicalizationMethod	*string*	Método de canonización del nodo `signedInfo`
signedPropertiesCanonicalizationMethod	*string*	Método de canonización del nodo `signedProperties`
xPathLocationString	*string*	Selector del nodo bajo el que se insertará la firma, en formato XPath: **Nodos sin namespace** Por ejemplo `<factura>` podríamos emplear: `"xPathLocationString": "//factura"` Referencia: - `https://www.w3schools.com/xml/xpath_syntax.asp` **Nodos con namespace** por ejemplo: `<T:TicketBai xmlns:T="urn:ticketbai:emision">` , podríamos emplear: `"xPathLocationString": "/*[\"TicketBai\"=local-name()]/Sujetos/Emisor"` para seleccionar el nodo Emisor Referencia: - `https://www.ibm.com/support/pages/how-use-xpath-namespaces-rit`
claimedSignerRoles	*array*	Roles del firmante
transformAlgorithms	*array*	Algoritmos de transformación para los nodos. Posibles valores: - `"http://www.w3.org/TR/2001/REC-xml-c14n-20010315"` - `"http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments"` - `"http://www.w3.org/2001/10/xml-exc-c14n#"` - `"http://www.w3.org/2001/10/xml-exc-c14n#WithComments"` - `"http://www.w3.org/2006/12/xml-c14n11"` - `"http://www.w3.org/2006/12/xml-c14n11#WithComments"` - `"http://santuario.apache.org/c14n/physical"`
dssReferenceUri	*string*	Identificador del nodo a firmar

Configuración TSA

Para los tipos de firma que incluyan sellado de tiempo, se debe informar la configuración de la TSA.

```
{
  "url": "http://tsa.example.com/",
  "user": "tsa_user",
  "password": "tsa_pass",
  "type": "USER",
  "certificateCode": "tsa_certificate_code"
}
```

Parámetro	Tipo	Descripción
type	*string*	Tipo de TSA. Si requiere autenticación con usuario y password, usaremos el valor `USER`, si requiere autenticación con certificado `CERTIFICATE`, si requiere autenticación con certificado por TSL `CERTIFICATE_TLS`, si no, el valor `URL`
user	*string*	Usuario para la autenticación en la TSA(solo para `type` con valor `USER`)
password	*string*	Contraseña para la autenticación en la TSA(para los `type` con valor `USER` o `CERTIFICATE_TLS`)
url	*string*	URL de de la TSA
certificateCode	*string*	Código del certificado para la autenticación en la TSA(para los `type` con valor `CERTIFICATE` o `CERTIFICATE_TLS`)

Configuración de políticas

Para que la firma tenga una política y se la pueda considerar de tipo EPES, podemos definir los valores de la misma con esta configuración.

```
{
  "id": "102039485-10283757-102837575",
  "description": "Sample policy",
  "digestAlgorithm": "SHA256",
  "digestValueB64": "JVBERi0xLjMKJcTl8uXlRU9GC",
  "url" : "https://sample/lorem_ipsum_dolor_sit_amet.pdf",
  "contentHintsDescription": "Lorem ipsum dolor sit amet",
  "contentHintsType": "Lorem ipsum dolor sit amet"
}
```

Parámetro	Tipo	Descripción
id	*string*	Identificador de la política
description	*string*	Descripción de la política
digestAlgorithm	*string*	Algoritmo de cifrado. Posibles valores: - `SHA1` - `SHA224` - `SHA256` - `SHA384` - `SHA512` - `RIPEMD160` - `MD2` - `MD5`
digestValueB64	*string*	Valor (codificado en Base64)
url	*string*	El SpURI (calificador de política de firma). El calificador spURI contendrá un valor de URL donde se puede obtener una copia del documento de política de firma.
contentHintsDescription	*string*	Descripción de la ayuda
contentHintsType	*string*	Tipo de ayuda

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
{
  "authCode": "1aeb979ddcf247e9ad46ee73e19a326d",
  "exeCode": "f116305e7f7c44f3a29385028c5374ba"
}
```

Donde:

Parámetro	Tipo	Descripción
authCode	*string*	Código de autorización
exeCode	*string*	Código de ejecución

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_request	Petición incorrecta. Alguno de los parámetros de entrada no es correcto. (HTTP Status: 400)
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
user_not_found	El usuario no es correcto o no está activo (HTTP Status: 404)

Autenticación del usuario en Viafirma Fortress y selección de certificado

En la sección Autenticación del usuario y autorización de operaciones de firma, se describe el procedimiento de autenticación y selección de certificado empleados en el proceso de firma.

Como resultado del proceso, se actualizará la operación de preparación de firma, asignando el certificado, y autorizando la operación para poder firmar los documentos.

Ejecución de la firma

Con este método firmaremos los documentos asociados a la operación de preparación de firma.

Uso del servicio

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature/{executionCode}/execute`

Además, en la cabecera HTTP de la petición `POST` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature/f116305e7f7c44f3a29385028c5374ba/execute`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `executionCode` : Código de ejecución que devolvió el método signature

Ejemplo:

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature/f116305e7f7c44f3a29385028c5374ba/execute`

Parámetros del servicio

Este servicio recibe por parámetros el código de ejecución executionCode, resultante del procedimiento de preparación de la firma, con dicho código obtendremos la información asociada a cada operación de firma.

Los parámetros que se reciben (en formato `application/json`) vacío:

```
{
}
```

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
[
  {
    "bytesB64": "a910b000d4f1a2b...",
    "signatureCode": "e2470412-33cc-467a-b357-880fe621920f",
    "mimeType": "application/pdf",
    "ref": "#1"
  },
  ...
]
```

Donde:

Parámetro	Tipo	Descripción
bytesB64	*string*	Documento firmado, codificado en Base64. Si se especifica el formato de firma asíncrona el resultado no incluye el documento firmado. Será necesario invocar al servicio de descarga del documento usando el `signatureCode` obtenido.
signatureCode	*string*	Identificador de firma
mimeType	*string*	MIME type del documento firmado
ref	*string*	Se devuelve el mismo valor si fue informado en la solicitud de firma

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_request	Petición incorrecta. Alguno de los parámetros de entrada no es correcto. (HTTP Status: 400)
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)

not_authorized_signature	El usuario no ha sido autenticado en los factores de autenticación (HTTP Status: 401)
user_not_found	El usuario no es correcto o no está activo (HTTP Status: 404)
certificate_not_found	El certificado con el que se desea firmar no es correcto o no está activo (HTTP Status: 404)
signature_error	Error durante la firma (HTTP Status: 500)

Obtención de documento firmado

Con este método podemos obtener un documento firmado usando un identificador de firma.

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/signature/download/{signature_code}`

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `signature_code` : Código de la firma de la que se desea obtener el documento

Ejemplo:

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/signature/download/C0XJ-XOAK-OF1O-TYJ7-S164-3197-3571-05`

Respuesta del servicio

La respuesta de este servicio vendrá dada en formato `application/octet-stream` .

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error

document_not_found	No se ha encontrado ningún documento para el ID de firma provisto (HTTP Status: 404)

Descarga de documento firmado

Con este método podemos descargar un documento firmado usando un identificador de firma.

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/signature/download/{signature_code}`

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `signature_code` : Código de la firma de la que se desea descargar el documento

Ejemplo:

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/signature/download/C0XJ-XOAK-OF1O-TYJ7-S164-3197-3571-05`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/octet-stream`)

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
document_not_found	No se ha encontrado ningún documento para el ID de firma provisto (HTTP Status: 404)

API: Métodos relacionados con la firma desatendida con certificado

```
Última actualización: 13 noviembre 2024
```

El proceso de firma desatendida en Viafirma Fortress, constará de los procesos de:

- Autenticación del cliente
- Solicitud de firma
- Ejecución de la firma
- Obtención del documento/s firmado/s

En los siguientes apartados describiremos los métodos disponibles en Viafirma Fortress, asociados a las operaciones de firma:

Nota: Para acceder a estos métodos es necesario tener un **token de acceso** (`access_token`) obtenido mediante una petición de autenticación y autorización con un `scope` de tipo `client` y un `grant_type` de tipo `client_credentials` , para lo que hay que seguir los pasos indicados en esta sección de la documentación.

Solicitud de firma desatendida

Uso del servicio

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Además, en la cabecera HTTP de la petición `POST` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Parámetros del servicio

Este servicio recibe por parámetros la configuración de la firma empleada por cada documento a firmar, donde se indica, entre otras cosas, el tipo de firma que se quiere realizar, el documento a firmar…

Los parámetros que se reciben (en formato `application/json`) tienen la siguiente forma:

```
{
  "certificateCode": "b8a25e04ab864583bb5ea8d02883e832",
  "signatureConfigurations": [
    {
      "ref": "#1",
      "document": {
```

```
        "bytesB64": "JVBERi0xLjMKJcTl8uXrp/Og0MTGCjQ...",
        "name": "contrato.pdf"
      },
      "signatureType": "PADES_B",
      "signatureAlgorithm": "RSA_SHA256",
      "packaging": "ENVELOPED",
      "padesConfiguration": {
        "stamper": {
          "csvPath": "http://localhost:7080/fortress/v#",
          "logoB64": "iVBORw0KGgoAAAANSUhEUgAAAWYAAABsCAYAAABZyhj...",
          "page": 1,
          "type": "QR_BARCODE128",
          "xAxis": 80,
          "yAxis": 700
        }
      }
    }
  ]
}
```

Nota: Los detalles de los parámetros `padesConfiguration`, `xadesConfiguration`, `tsa` y `policy` se muestran más adelante.

Donde:

Parámetro	Tipo	Descripción
certificateCode	*string*	Código del certificado a emplear en la firma, se debe consultar en la pestaña **Certificados** del apartado **Configuración** del detalle del sistema cliente o grupo donde se aloje el certificado
certificatePassword	*string*	Contiene la password del certificado, este campo solamente está permitido para la firma desatendida.
async	*string*	Si se establece con el valor `true` la llamada al servicio de ejecución de firma se realiza de forma asíncrona.
callbackUrl	*string*	Si se informa una URL y la firma se realiza de forma asíncrona, al finalizar la firma Fortress realiza una petición POST a dicha URL con el estado final de ejecución.
signatureConfigurations/document/name	*string*	Nombre del documento a firmar
signatureConfigurations/document/bytesB64	*string*	Documento a firmar, codificado en Base64
signatureConfigurations/document/url	*string*	URL del documento a firmar
signatureConfigurations/signatureType	*string*	Tipo de firma. Valores disponibles: - CADES_B - CADES_T - CADES_LT - CADES_LTA - PADES_B - PADES_T - PADES_LT - PADES_LTA - XADES_B - XADES_T - XADES_LT - XADES_LTA - PKCS1
signatureConfigurations/signatureAlgorithm	*string*	Algoritmo que se usará para cifrar la firma. Valores disponibles: - `RSA_SHA1` - `RSA_SHA224` - `RSA_SHA256` - `RSA_SHA384` - `RSA_SHA512`

signatureConfigurations/packaging	*string*	Envoltura de la firma. Valores disponibles: - `ENVELOPED` - `ENVELOPING` - `DETACHED`
signatureConfigurations/reason	*string*	OPCIONAL, motivo de la firma
signatureConfigurations/location	*string*	OPCIONAL, localización de la firma
signatureConfigurations/ref	*string*	Si se informa, se devolverá el mismo valor en el resultado de la firma.

Configuración PAdES

Esta configuración solo se aplica para las firmas cuyo `signatureType` es de tipo PAdES (PAdES B, PAdES T, PAdES LT, PAdES LTA).

```
"padesConfiguration": {
    "stamper": { }
}
```

El objeto stamper es opcional, y sirve para definir un sello visual asociado a la firma PAdES.

```
{
  "stamper": {
    "csvPath": "https://fortress.viafirma.com/fortress/v#",
    "imageB64": "JVBERi0xLjMKJcTl8uXlRU9GC...",
    "logoB64": "JVBERi0xLjMKJcTl8uXlRU9GC...",
    "page": 1,
    "rotation": "ROTATE_90",
    "textLine1": "Sample line 1",
    "textLine2": "Sample line 2",
    "textLine3": "Sample line 3",
    "type": "QR_BARCODE128",
    "xAxis": 100,
    "yAxis": 100,
    "timeZoneId": "America/Santo_Domingo"
  }
}
```

Parámetro	Tipo	Descripción
stamper/csvPath	*string*	URL de verificación de la firma
stamper/xAxis	*int*	Posición (en el eje X) del sello de firma
stamper/yAxis	*int*	Posición (en el eje Y) del sello de firma
stamper/imageB64	*string*	Imagen de fondo del sello de firma
stamper/logoB64	*string*	Logo del sello de firma (se pintará en la parte inferior derecha del sello)
stamper/page	*int*	Página donde se pintará el sello. Se puede usar el valor `-1` para indicar la última página o el valor `0` para pintar el sello en todas las páginas del documento
stamper/rotation	*string*	Rotación del sello. Si se informa, el sello rotará los grados indicados. Valores posibles: - `ROTATE_90` - `ROTATE_270`
stamper/textLine1	*string*	Primera linea textual a pintar en el contenido del sello de firma
stamper/textLine2	*string*	Segunda linea textual a pintar en el contenido del sello de firma
stamper/textLine3	*string*	Tercera linea textual a pintar en el contenido del sello de firma
		Tipo de sello. Valores disponibles:

stamper/type	*string*	- `PDF417` - `QR_BARCODE128` - `QR` - `BARCODE128` - `IMAGE` - `TEXT`
stamper/timeZoneId	*string*	String que se corresponderá con la lista estándar de zonas horarias.

Dependiendo del `type` elegido, el sello tendrá unas dimensiones preestablecidas (en píxeles):

- `PDF417` : 300x130
- `QR_BARCODE128` : 600x100
- `QR` : 450x50
- `BARCODE128` : 550x70
- `IMAGE` : Mantiene las dimensiones de la imagen especificada en `imageB64`
- `TEXT` : 400x50

Si en llamada a la API NO viene informado el valor `timeZoneId` aplicaremos el siguiente criterio:

- En caso de firma desatendida, si el Sistema Cliente pertenece a algún grupo que lo tenga informado, aplicamos el del primer grupo que lo tenga informado, si no aplicamos el configurado por defecto en el sistema.

Configuración XAdES

Esta configuración solo se aplica para las firmas cuyo `signatureType` es de tipo XAdES (XAdES B, XAdES T, XAdES LT, XAdES LTA)

```
{
  "signedInfoCanonicalizationMethod": "http://www.w3.org/TR/2001/REC-xml-c14n-20010315",
  "signedPropertiesCanonicalizationMethod": "http://www.w3.org/TR/2001/REC-xml-c14n-20010315",
  "xPathLocationString": "//book[@id='bk101-1']",
  "claimedSignerRoles": [
    "role1",
    "role2"
  ],
  "transformAlgorithms": [
    "http://www.w3.org/TR/2001/REC-xml-c14n-20010315"
  ],
  "dssReferenceUri": "http://dsa-reference.example.com/"
}
```

Donde:

Parámetro	Tipo	Descripción
signedInfoCanonicalizationMethod	*string*	Método de canonización del nodo `signedInfo`
signedPropertiesCanonicalizationMethod	*string*	Método de canonización del nodo `signedProperties`
xPathLocationString	*string*	Selector del nodo bajo el que se insertará la firma, en formato XPath
claimedSignerRoles	*array*	Roles del firmante
transformAlgorithms	*array*	Algoritmos de transformación para los nodos. Posibles valores: - `"http://www.w3.org/TR/2001/REC-xml-c14n-20010315"` - `"http://www.w3.org/TR/2001/REC-xml-c14n-20010315#WithComments"` - `"http://www.w3.org/2001/10/xml-exc-c14n#"` - `"http://www.w3.org/2001/10/xml-exc-c14n#WithComments"` - `"http://www.w3.org/2006/12/xml-c14n11"` - `"http://www.w3.org/2006/12/xml-c14n11#WithComments"` - `"http://santuario.apache.org/c14n/physical"`
dssReferenceUri	*string*	Identificador del nodo a firmar

Configuración TSA

Para los tipos de firma que incluyan sellado de tiempo, se debe informar la configuración de la TSA.

```
{
  "url": "http://tsa.example.com/",
  "user": "tsa_user",
  "password": "tsa_pass",
  "type": "USER",
  "certificateCode": "tsa_certificate_code"
}
```

Parámetro	Tipo	Descripción
type	*string*	Tipo de TSA. Si requiere autenticación con usuario y password, usaremos el valor `USER`, si requiere autenticación con certificado `CERTIFICATE`, si requiere autenticación con certificado por TSL `CERTIFICATE_TLS`, si no, el valor `URL`
user	*string*	Usuario para la autenticación en la TSA (solo para `type` con valor `USER`)
password	*string*	Contraseña para la autenticación en la TSA (para los `type` con valor `USER` o `CERTIFICATE_TLS`)
url	*string*	URL de de la TSA
certificateCode	*string*	Código del certificado para la autenticación en la TSA (para los `type` con valor `CERTIFICATE` o `CERTIFICATE_TLS`)

Configuración de políticas

Para que la firma tenga una política y se la pueda considerar de tipo EPES, podemos definir los valores de la misma con esta configuración.

```
{
  "id": "102039485-10283757-102837575",
  "description": "Sample policy",
  "digestAlgorithm": "SHA256",
  "digestValueB64": "JVBERi0xLjMKJcTl8uXlRU9GC",
  "url" : "https://sample/lorem_ipsum_dolor_sit_amet.pdf",
  "contentHintsDescription": "Lorem ipsum dolor sit amet",
  "contentHintsType": "Lorem ipsum dolor sit amet"
}
```

Parámetro	Tipo	Descripción
id	*string*	Identificador de la política
description	*string*	Descripción de la política
digestAlgorithm	*string*	Algoritmo de cifrado. Posibles valores: - `SHA1` - `SHA224` - `SHA256` - `SHA384` - `SHA512` - `RIPEMD160` - `MD2` - `MD5`
digestValueB64	*string*	Valor (codificado en Base64)
url	*string*	El SpURI (calificador de política de firma). El calificador spURI contendrá un valor de URL donde se puede obtener una copia del documento de política de firma.

contentHintsDescription	*string*	Descripción de la ayuda
contentHintsType	*string*	Tipo de ayuda

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
{
  "authCode": "1aeb979ddcf247e9ad46ee73e19a326d",
  "exeCode": "f116305e7f7c44f3a29385028c5374ba"
}
```

Donde:

Parámetro	Tipo	Descripción
authCode	*string*	Código de autorización
exeCode	*string*	Código de ejecución

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_request	Petición incorrecta. Alguno de los parámetros de entrada no es correcto. (HTTP Status: 400)
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)

Ejecución de la firma

Con este método firmaremos los documentos asociados a la operación de preparación de firma.

Uso del servicio

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature/{executionCode}/execute`

Además, en la cabecera HTTP de la petición `POST` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `executionCode` : Código de ejecución que devolvió el método signature

Ejemplo:

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature/f116305e7f7c44f3a29385028c5374ba/execute`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Parámetros del servicio

Este servicio recibe por parámetros el código de ejecución executionCode, resultante del procedimiento de preparación de la firma, con dicho código obtendremos la información asociada a cada operación de firma.

Se debe proporcionar(en formato `application/json`) vacío:

```
{
}
```

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
[
  {
    "bytesB64": "a910b000d4f1a2b...",
    "signatureCode": "e2470412-33cc-467a-b357-880fe621920f",
    "mimeType": "application/pdf",
    "ref": "#1"
  },
  ...
]
```

Donde:

Parámetro	Tipo	Descripción
bytesB64	*string*	Documento firmado, codificado en Base64
signatureCode	*string*	Identificador de firma
mimeType	*string*	MIME type del documento firmado
ref	*string*	Se devuelve el mismo valor si fue informado en la solicitud de firma

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción

error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_request	Petición incorrecta. Alguno de los parámetros de entrada no es correcto. (HTTP Status: 400)
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
certificate_not_found	El certificado con el que se desea firmar no es correcto o no está activo (HTTP Status: 404)
invalid_certificate_password	La password del certificado es incorrecta (HTTP Status: 401)
locked_certificate	El certificado se encuentra bloqueado (HTTP Status: 401)
signature_error	Error durante la firma (HTTP Status: 500)

Descarga de documento firmado

Con este método podemos descargar un documento firmado usando un identificador de firma.

Uso del servicio

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/signature/download/{signature_code}`

Además, en la cabecera HTTP de la petición `GET` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress
- `signature_code` : Código de la firma de la que se desea descargar el documento

Ejemplo:

Método: `GET`
URL: `{viafirma_fortress_url}/api/v1/signature/download/C0XJ-XOAK-OF1O-TYJ7-S164-3197-3571-05`

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/octet-stream`)

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
document_not_found	No se ha encontrado ningún documento para el ID de firma provisto (HTTP Status: 404)

API: Método relacionado con la extensión de firma

El proceso de firma en Viafirma Fortress, constará de los procesos de:

- Autenticación del cliente
- Solicitud de extensión de firma
- Obtención del documento/s extendido/s firmado/s

En los siguientes apartados describiremos los métodos disponibles en Viafirma Fortress, asociado a las operaciones de extensión de firma:

Solicitud de firma

Nota: Para acceder a estos métodos es necesario tener un **token de acceso** (`access_token`) obtenido mediante una petición de autenticación y autorización con un `scope` de tipo `client` y un `grant_type` de tipo `client_credentials` , para lo que hay que seguir los pasos indicados en esta sección de la documentación.

Solicitud de extensión de firma

Uso del servicio

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/signature/extend`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Además, en la cabecera HTTP de la petición `POST` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `POST`
URL: `https://fortress.viafirma.com/fortress/api/v1/signature/extend`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Parámetros del servicio

Este servicio recibe por parámetros la configuración de la extensión de firma empleada por cada documento a extender, donde se indica, entre otras cosas, el tipo de firma al que se desea extended el documento firmado, el documento a extender …

Los parámetros que se reciben (en formato `application/json`) tienen la siguiente forma:

```
{
  "extendSignatureConfigurations": [
    {
      "document": {
        "bytesB64": "JVBERi0xLjMKJcTl8uXrp/Og0MTGCjQ...",
        "name": "contrato.pdf"
      },
```

```
      "signatureType": "PADES_LTA",
      "signatureAlgorithm": "RSA_SHA256",
      "packaging": "ENVELOPED",
      "tsa": {
        "type": "URL",
        "url": "https://testservices.viafirma.com/via-tsa/tsa"
      }
    }
  ]
}
```

Nota: Los detalles de los parámetros `padesConfiguration`, `xadesConfiguration`, `tsa` y `policy` se muestran más adelante.

Donde:

Parámetro	Tipo	Descripción
userCode	*string*	[OPCIONAL] Usuario que debe extender la firma, si el sistema cliente no informa dicho valor, Viafirma Fortress no solicitará el usuario a emplear en el proceso de extensión de firma
extendSignatureConfigurations/document/name	*string*	Nombre documento a extender
extendSignatureConfigurations/document/bytesB64	*string*	Documento a extender, codificado en Base64
extendSignatureConfigurations/document/url	*string*	Url ocumento a extender
extendSignatureConfigurations/signatureType	*string*	Tipo de firma. Valores disponibles: - `CADES_T` - `CADES_LT` - `CADES_LTA` - `PADES_T` - `PADES_LT` - `PADES_LTA` - `XADES_T` - `XADES_LT` - `XADES_LTA`
extendSignatureConfigurations/packaging	*string*	Envoltura de la firma. Valores disponibles: - `ENVELOPED` - `ENVELOPING` - `DETACHED`
extendSignatureConfigurations/original/name	*string*	[Solo para envolturas DETACHED] Nombre documento original a extender
extendSignatureConfigurations/original/bytesB64	*string*	[Solo para envolturas DETACHED] Documento original a extender, codificado en Base64
extendSignatureConfigurations/original/url	*string*	[Solo para envolturas DETACHED] Url documento original a extender

Configuración TSA

Para los tipos de firma que incluyan sellado de tiempo, se debe informar la configuración de la TSA.

```
{
  "url": "http://tsa.example.com/",
  "user": "tsa_user",
  "password": "tsa_pass",
  "type": "USER",
  "certificateCode": "tsa_certificate_code"
}
```

Parámetro	Tipo	Descripción

type	*string*	Tipo de TSA. Si requiere autenticación con usuario y password, usaremos el valor `USER`, si requiere autenticación con certificado `CERTIFICATE`, si requiere autenticación con certificado por TSL `CERTIFICATE_TLS`, si no, el valor `URL`
user	*string*	Usuario para la autenticación en la TSA(solo para `type` con valor `USER`)
password	*string*	Contraseña para la autenticación en la TSA(para los `type` con valor `USER` o `CERTIFICATE_TLS`)
url	*string*	URL de de la TSA
certificateCode	*string*	Código del certificado para la autenticación en la TSA(para los `type` con valor `CERTIFICATE` o `CERTIFICATE_TLS`)

Configuración de políticas

Para que la firma tenga una política y se la pueda considerar de tipo EPES, podemos definir los valores de la misma con esta configuración.

```
{
  "id": "102039485-10283757-102837575",
  "description": "Sample policy",
  "digestAlgorithm": "SHA256",
  "digestValueB64": "JVBERi0xLjMKJcTl8uXlRU9GC",
  "contentHintsDescription": "Lorem ipsum dolor sit amet",
  "contentHintsType": "Lorem ipsum dolor sit amet"
}
```

Parámetro	Tipo	Descripción
id	*string*	Identificador de la política
description	*string*	Descripción de la política
digestAlgorithm	*string*	Algoritmo de cifrado. Posibles valores: - `SHA1` - `SHA224` - `SHA256` - `SHA384` - `SHA512` - `RIPEMD160` - `MD2` - `MD5`
digestValueB64	*string*	Valor (codificado en Base64)
contentHintsDescription	*string*	Descripción de la ayuda
contentHintsType	*string*	Tipo de ayuda

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
{
  "ref": "d8e3d98dc20e46188fd067df28048934",
  "bytesB64": "MIMBKM8GCSqGSIb3DQEHAqCDASi/MIMBKLoCAQUxDzANBglghkgBZQMEAgEFADCC1QsGCSqGSIb3DQEHAaCC1PwEgtT4JVBERi0xLjMKJcTl8uX
rp..."
}
```

Donde:

Parámetro	Tipo	Descripción
ref	*string*	referencia de la extensión

bytesB64	*string*	Base 64 del documento extendido

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente aspecto:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_request	Petición incorrecta. Alguno de los parámetros de entrada no es correcto. (HTTP Status: 400)
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)

API: Métodos relacionados con la encriptación / desencriptación

```
Última actualización: 05 abril 2021
```

Nota: Para acceder a estos métodos es necesario tener un **token de acceso** (`access_token`) obtenido mediante una petición de autorización con un `scope` de tipo `client` y un `grant_type` de tipo `client_credentials` , para lo que hay que seguir los pasos indicados en la sección Autenticación del sistema Cliente de la documentación.

Solicitud de Encriptación / Desencriptación

Permite encriptar / desencriptar mediante RSA los bytes indicados, usando un certificado asociado a un sistema cliente o grupo.

Uso del servicio

Método: `POST` URL: `{viafirma_fortress_url}/api/v1/encrypt`

Donde:

- `viafirma_fortress_url` : URL base de la implementación de Viafirma Fortress, por ejemplo https://sandbox.viafirma.com/fortress o https://fortress.viafirma.com/fortress

Además, en la cabecera HTTP de la petición `POST` debe incluirse el token de acceso (`access_token`) de la siguiente forma:

```
Authorization: Bearer {access_token}
```

Ejemplo:

Método: `POST`
URL: `{viafirma_fortress_url}/api/v1/encrypt`
Header de la petición: `Authorization: Bearer 0b79bab50daca910b000d4f1a2b675d604257e42`

Parámetros del servicio

Los parámetros que se reciben (en formato `application/json`) tienen la siguiente forma:

```
{
  "certificateCode": "580fe337eba1483683290cbbf94982a3",
  "mode": "ENCRYPT",
  "bytesB64": "dGVzdA=="
}
```

Donde:

Parámetro	Tipo	Descripción
certificateCode	*string*	Código del certificado de sistema cliente o grupo a usar en la encriptación / desencriptación. Para la encriptación se usará la clave pública del certificado mientras que para la desencriptación se usa la clave privada.
mode	*string*	Indica si se trata de una operación de encriptado o desencriptado, los valores posibles son `ENCRYPT` , `DECRYPT` .

bytesB64	*string*	Contenido a encriptar / desencriptar en formato Base64.

Respuesta del servicio

La respuesta de este servicio vendrá dada (en formato `application/json`) de la siguiente forma:

```
{
  "bytesB64": "ggj5mRTVh3FKAz4wf2EmaX7Zfr...=="
}
```

Donde:

Parámetro	Tipo	Descripción
bytesB64	*string*	Resultado de la encriptación / desencriptación en formato Base64.

Errores del servicio

Los errores devueltos por el servicio (devueltos en formato `application/json`) tienen el siguiente formato:

```
{
  "error": "error_code",
  "error_description": "error_description"
}
```

Donde:

Parámetro	Tipo	Descripción
error	*string*	Código del error
error_description	*string*	Descripción del error

Posibles errores:

Código de error	Error
invalid_request	Petición incorrecta. Alguno de los parámetros de entrada no es correcto. (HTTP Status: 400)
invalid_token	El `access_token` utilizado no es correcto (HTTP Status: 401)
certificate_not_found	El certificado especificado no es correcto o no está activo (HTTP Status: 404)

Ejemplos rápidos de integración

Nota: Todas las referencias a ficheros o documentos codificados con Base64 se muestran truncadas para facilitar la lectura de esta documentación.

Autenticación de usuario, operaciones de consulta

La aplicación tercera "SAMPLE APP" quiere autenticar a un usuario para consultar los datos del usuario, cuyo código es `sample_user`.

Requisitos previos:

- La aplicación debe estar dada de alta como sistema cliente en Viafirma Fortress
- Se le debe haber provisto de un `client_id`. En este ejemplo será `sample_app`
- Se le debe haber provisto de un `client_secret`. En este ejemplo será `12345`
- Se le debe haber configurado una URL de retorno permitida: `http://www.example.com/auth`

En el momento en que la aplicación "SAMPLE APP" quiera realizar la autenticación del usuario contra Viafirma Fortress, lo redireccionará a una URL:

```
{viafirma_fortress_url}/oauth2/v1/auth?
scope=profile&
state=&
redirect_uri=http://www.example.com/auth&
response_type=code&
client_id=sample_app&
user_code=sample_user
```

En esta URL se le presentarán al usuario los distintos factores de autenticación de Viafirma Fortress en los que esté enrolado. Utilizará alguno de ellos para autenticarse y autorizar la operación. Una vez finalizado el proceso, Viafirma Fortress devolverá el control a la aplicación "SAMPLE APP", redirigiendo a la URL de retorno: `http://www.example.com/auth?state=&code=e2470412-33cc-467a-b357-880fe621920f`

Aesta URL se le enviará como parámetro de URL el valor del **código de autorización**, con el cual podrá solicitar un **token de acceso** con el que operar (p. ej. obtener información sobre el estado del usuario).

Para obtener ese **token de acceso**, la aplicación "SAMPLE APP" realizará una petición a Viafirma Fortress:

- Método: `POST`
- URL: `https://fortress.viafirma.com/fortress/oauth2/v1/token`
- Parámetros:
  - `code`: Cuyo valor es el código de autorización obtenido previamente: `"e2470412-33cc-467a-b357-880fe621920f"`
  - `client_id`: Cuyo valor es el determinado en Viafirma Fortress para la aplicación "SAMPLE APP": `"sample_app"`
  - `client_secret`: Cuyo valor es el determinado en Viafirma Fortress para la aplicación "SAMPLE APP": `"12345"`
  - `redirect_uri`: Cuyo valor es la URL de retorno para la que se hizo la petición de autorización: `"http://www.example.com/auth"`
  - `grant_type`: Este valor es fijo: `"authorization_code"`

El resultado de esta petición `POST` será:

```
{
  "access_token": "1/fFAGRNJru1FTz70BzhT3Zg",
  "expires_in": 3920,
  "token_type": "Bearer"
}
```

Una vez obtenidos estos valores, podemos considerar que el usuario ha sido autenticado correctamente. Podremos, además, usar el valor de `access_token` para realizar operaciones de consulta sobre la API de Viafirma Fortress (p. ej. obtener el estado del usuario, los certificados de un usuario "**scope=CERTIFICATES**"o el detalle de un certificado "**scope=CERTIFICATE**").

Firma de un documento PDF

La aplicación tercera "SAMPLE APP" quiere que el usuario `sample_user` firme un documento PDF.

Requisitos previos:

- La aplicación debe estar dada de alta como sistema cliente en Viafirma Fortress
- Se le debe haber provisto de un `client_id` . En este ejemplo será `sample_app`
- Se le debe haber provisto de un `client_secret` . En este ejemplo será `12345`
- Se le debe haber configurado una URL de retorno permitida: `http://www.example.com/sign`

Obtención token de cliente

En el momento en que la aplicación "SAMPLE APP" quiera comenzar la operación de firma del documento PDF, deberá obtener un token de sistema cliente.

Para obtener ese **token de acceso**, la aplicación "SAMPLE APP" realizará una petición a Viafirma Fortress:

- Método: `POST`
- URL: `https://fortress.viafirma.com/fortress/oauth2/v1/token`
- Parámetros:
  - `client_id` : Cuyo valor es el determinado en Viafirma Fortress para la aplicación "SAMPLE APP": `"sample_app"`
  - `client_secret` : Cuyo valor es el determinado en Viafirma Fortress para la aplicación "SAMPLE APP": `"12345"`
  - `redirect_uri` : Cuyo valor es la URL de retorno para la que se hizo la petición de autorización: `"http://www.example.com/auth/response"`
  - `grant_type` : Este valor es fijo: `"client_credentials"`

```
https://fortress.viafirma.com/fortress/oauth2/v1/token?
grant_type=client_credentials&
redirect_uri=http://www.example.com/auth/response&
client_id=sample_app&
client_secret=12345
```

El resultado de esta petición `POST` será:

```
{
  "access_token": "666b3b58ecb54db784e2eafdfc66e113",
  "expires_in": 3920,
  "token_type": "Bearer"
}
```

Solicitud de Firma

Con el access_token resultante de la llamada, el sistema cliente llamará al método signature:

Método: `POST`
URL: `https://fortress.viafirma.com/fortress/api/v1/signature`
Header de la petición: `Authorization: Bearer 666b3b58ecb54db784e2eafdfc66e113`

```
{
  "userCode":"abcde",
  "redirectUri":"http://localhost:8080/ fortress -demo/ sign",
  "signatureConfigurations": [
```

```
    {
      "signatureType":"PADES_B",
      "signatureAlgorithm":"RSA_SHA256",
      "packaging":"ENVELOPED",
      "document": {
        "name":"example.pdf",
        "bytesB64":"JVBERi0xLjMKJcTl8uXrp/Og0MTGCjQgMCBvYmoKPDwgL0xlbmd0aC..."
      },
      "padesConfiguration": {
        "stamper": {
          "csvPath":"http://localhost:7080/fortress/v#",
          "logoB64":"iVBORw0KGgoAAAANSUhEUgA...",
          "page": 1,
          "type":"QR_BARCODE128",
          "xAxis": 80,
          "yAxis": 700
        }
      }
    }
  ]
}
```

En el cuerpo del método el sistema debe incluir un json con el siguiente formato:

- userCode: código de usuario
- redirectUri: Uri donde debe redirigir la operación una vez finalizada
- signatureConfigurations: por cada documento a firmar, se deberá indicar el documento, el tipo de firma y las políticas de firma.

El resultado de esta petición `POST` será:

```
{
  "authCode": "d8e3d98dc20e46188fd067df28048934",
  "exeCode": "cae2c9fe4f4b41888d42ac18a88096a2"
}
```

Autorización de la solicitud de firma

En el momento en que la aplicación "SAMPLE APP" quiera comenzar la operación de firma del documento PDF, redireccionará al usuario a una URL para que autorice la operación de firma y seleccione el certificado a emplear:

```
https://sandbox.viafirma.com/fortress/oauth2/v1/auth?signature_code=7b3e77ad2aef4e479c2ae39f497cfe0c&scope=signature&client_id
=fortress-dem&redirect_uri=https%3A%2F%2Fsandbox.viafirma.com%2fortress-demo%2Fsign%2Fresponse
```

En esta URL se le presentarán al usuario los distintos factores de autenticación de Viafirma Fortress en los que esté enrolado. Utilizará alguno de ellos para autenticarse y autorizar la operación de firma. Una vez autenticado, se le mostrarán los distintos certificados que Viafirma Fortress esté custodiando para este usuario, para que seleccione con cuál quiere realizar la firma.

Ejecutar Firma

Una vez obtenidos estos valores, podemos considerar que el usuario ha sido autenticado correctamente y ha autorizado la operación de firma, con lo que se podrá llamar al servicio de firma. Para ello, se realiza una petición a Viafirma Fortress, incluyendo el token de acceso y el identificador de certificado obtenidos en el paso anterior:

- Método HTTP: `POST`
- URL: `https://fortress.viafirma.com/fortress/api/v1/signature/cae2c9fe4f4b41888d42ac18a88096a2/execute` Header de la petición: `Authorization: Bearer 666b3b58ecb54db784e2eafdfc66e113`

La respuesta de este servicio será:

```
{
  "documentB64": "LjMKJcTl8u...",
  "mimeType": "application/pdf",
  "signatureCode": "TFOR-TRES-SOAK-OF1O-TXFR-5151-8007-9109-77"
}
```

En el atributo `documentB64` tendremos el documento firmado (codificado en Base64), y en `signatureCode` el identificador de firma.

Extensión de Firma

Con el access_token resultante de la llamada, el sistema cliente llamará al método extend:

Método: `POST`
URL: `https://fortress.viafirma.com/fortress/api/v1/signature/extend`
Header de la petición: `Authorization: Bearer 666b3b58ecb54db784e2eafdfc66e113`

```
{
  "extendSignatureConfigurations": [
    {
      "document": {
        "bytesB64": "JVBERi0xLjMKJcTl8uXrp/Og0MTGCjQ...",
        "name": "contrato.pdf"
      },
      "signatureType": "PADES_LTA",
      "signatureAlgorithm": "RSA_SHA256",
      "packaging": "ENVELOPED",
      "tsa": {
        "type": "URL",
        "url": "https://testservices.viafirma.com/via-tsa/tsa"
      }
    }
  ]
}
```

En el cuerpo del método el sistema debe incluir un json con el siguiente formato:

- userCode: código de usuario
- redirectUri: Uri donde debe redirigir la operación una vez finalizada
- extendSignatureConfigurations: por cada documento a firmar, se deberá indicar el documento, el tipo de firma y las políticas de firma.

El resultado de esta petición `POST` será:

```
{
  "ref": "d8e3d98dc20e46188fd067df28048934",
  "bytesB64": "MIMBKM8GCSqGSIb3DQEHAqCDASi/MIMBKLoCAQUxDzANBglghkgBZQMEAgEFADCC1QsGCSqGSIb3DQEHAaCC1PwEgtT4JVBERi0xLjMKJcTl8uXrp..."
}
```

Aplicación de ejemplo

Hemos desarrollado una aplicación de ejemplo que utiliza los principales servicios ofrecidos por **Viafirma Fortress**:

- Obtener el estado de un usuario (si tiene certificados y se encuentra enrolado en algún Factor de autenticación)
- Autenticación de un usuario.
- Solicitud de autorización para realizar la firma.
- Firma PAdES, XAdES.
- Firma en bucle.

Pulse aquí para descargar los fuentes

Requisitos

- JDK 1.7 o superior
- Maven 3.0+

Solicitar credenciales al departamento comercial:

- `client_id`. En este ejemplo será `sample_app`
- `client_secret`. En este ejemplo será `12345`

Cómo ejecutar la aplicación

El ejemplo usa Spring Boot para simplificar el arranque, ya que permite ejecutar la aplicación usando un servidor Tomcat embebido.

- Puedes importar la aplicación en tu IDE favorito y ejecutar la clase `com.viafirma.fortress.demo.FortressDemoApplication`.
- Una vez importado deberá configurar las credenciales proporcionadas en el fichero fortress-demo.properties alojado en /src/main/resource, por ejemplo:

```
fortress.demo.api.url=https://sandbox.viafirma.com/fortress/
fortress.demo.api.client_id=sample_app
fortress.demo.api.client_secret=12345
```

Una vez configurado, podrá:

- Iniciar la aplicación con el comando `mvn spring-boot:run`.
- Compilar la aplicación con `mvn clean package` y desplegar el WAR en un contenedor Tomcat o ejecutarlo directamente:

```
java -jar target/viafirma-fortress-demo.war
```

Cómo probar la aplicación

Una vez iniciada la aplicación, puedes acceder abriendo la siguiente URL en su explorador:

```
http://localhost:8080/fortress-demo/
```

Pantalla de login

Esta pantalla simula el login en la aplicación cliente, solo se utiliza para obtener el código de usuario a consultar en Viafirma Fortress.

Puedes acceder con el usuario `12345678Z` y cualquier clave.

Pantalla de opciones

Esta pantalla muestra cómo obtener la información básica de un usuario (si tiene certificados asociados y si está enrolado en algún Factor de autenticación)

También permite probar la autenticación y distintas posibilidades de firma.

Autenticación

Se puede autenticar al usuario de pruebas `12345678Z` usando el factor de autenticación PIN `1234` y el factor de autenticación OTP escaneando el siguiente QR con Google Authenticator:

Viafirma Fortress Desktop para Windows

Viafirma Fortress cuenta con un cliente (CSP), válido para Windows 7, 8 y 10 con el que podrás hacer uso de tus certificados centralizados en fortress de forma universal con cualquier otro sitio web o aplicación que lo requiera.

Versión para entorno de producción

Puedes encontrar el enlace de descarga en la web oficial:

- https://fortress.viafirma.com/

Versión para otros entornos

Esta versión está pensada para integradores y en general para poder conectar Viafirma Fortress Desktop con otro backend distinto al de producción mediante una opción de configuración que te permite introducir la URL deseada.

Puedes descargarlo desde el siguiente enlace según arquitectura de su sistema operativo:

- Viafirma Fortress Desktop para INTEGRADORES 64 bits
- Viafirma Fortress Desktop para INTEGRADORES 32 bits