API de assinatura digital GovBR

Este documento visa detalhar a estrutura da API REST para assinatura digital usando certificados avançados gov.br.

A API adota o uso do protocolo OAuth 2.0 para autorização de acesso e protocolo HTTP para acesso aos endpoints. Assim sendo, o uso da API envolve duas etapas:

1. Geração do token de acesso OAuth (Access Token)
2. Acesso ao serviço de assinatura

Geração do Access Token

Para geração do Access Token é necessário redirecionar o navegador do usuário para o endereço de autorização do servidor OAuth, a fim de obter seu consentimento para o uso de seu certificado para assinatura. Nesse processo, a aplicação deve usar credenciais previamente autorizadas no servidor OAuth. As seguintes credencias podem ser usadas para testes:

```
Servidor OAuth = https://sistemas.homologacao.ufsc.br/govbr/oauth2.0
Client ID= devLocal
Secret = younIrtyij3
URI de redirecionamento = http://127.0.0.1:*/**
```

As credenciais para Client ID “devLocal” estão configuradas no servidor OAuth para aceitar qualquer aplicação executando localmente (host 127.0.0.1, qualquer porta, qualquer caminho). Aplicações remotas não poderão usar essas credenciais de teste.

A URL usada para redirecionar o usuário para o formulário de autorização, conforme a especificação do OAuth 2.0, é a seguinte:

```
https://<Servidor OAuth>/authorize?response_type=code
                                  &redirect_uri=<URI de redirecionamento>
                                  &scope=sign
                                  &client_id=<clientId>
```

Nesse endereço, o servidor OAuth autentica o usuário e pede a autorização expressa do mesmo para acessar seu certificado para assinatura. Após a autorização ser dada pelo usuário, o servidor OAuth redireciona o mesmo para o endereço <URI de redirecionamento> especificado, e passa, como um parâmetro de query, o atributo Code. O <URI de redirecionamento> deve ser um endpoint da aplicação correspondente ao padrão autorizado no servidor OAuth, e capaz de receber e tratar o parâmetro “code”. Este atributo deve ser usado na fase seguinte do protocolo OAuth, pela aplicação, para pedir um Access Token ao servidor OAuth, com a seguinte requisição HTTP com método POST:

```
https://<Servidor OAuth>/token?code=<code>
                              &client_id=<clientId>
                              &grant_type=authorization_code
                              &client_secret=<secret>
                              &redirect_uri=<URI de redirecionamento>
```

O <URI de redirecionamento> deve ser exatamente o mesmo valor passado na requisição "authorize" anterior. O servidor OAuth retornará um objeto JSON contendo o Access Token, que deve ser usado nas requisições subsequentes aos endpoints do serviço.

Importante: O servidor OAuth está delegando a autenticação ao ambiente de **Staging** do gov.br

Obtenção do certificado do usuário

Para obtenção do certificado do usuário deve-se fazer uma requisição HTTP Get para o seguinte end-point:

```
https:// govbr-uws.homologacao.ufsc.br/CloudCertService/certificadoPublico
```

Deve-se enviar o cabeçalho Authorization com o tipo de autorização Bearer e o Access Token obtido anteriormente. Exemplo de requisição:

```
GET /CloudCertService/certificadoPublico HTTP/1.1
Host: govbr-uws.homologacao.ufsc.br
Authorization: Bearer <Access token>
```

Será retornado o certificado digital em formato PEM na resposta.

Realização da assinatura digital de um HASH SHA-256

Para assinar digitalmente um HASH SHA-256 usando a chave privada do usuário, deve-se fazer uma requisição HTTP POST para o seguinte end-point:

```
https:// govbr-uws.homologacao.ufsc.br/CloudCertService/assinarRaw
```

Deve-se enviar o cabeçalho Authorization com o tipo de autorização Bearer e o Access Token obtido anteriormente. Exemplo de requisição:

```
GET /CloudCertService/assinarRaw HTTP/1.1
Host: govbr-uws.homologacao.ufsc.br
Content-Type: application/json
Authorization: Bearer <Access token>
Content-Type: application/json

{"hashBase64":"<Hash SHA256 codificado em Base64>"}
```

Será retornada a assinatura digital SHA256-RSA codificada em Base64 na resposta.

Exemplo de aplicação

Anexo a este documento, encontra-se um pequeno exemplo PHP para prova de conceito.

Este exemplo é composto por 3 arquivos

- index.php → Formulário para upload de um arquivo

- upload.php → Script para recepção de arquivo e cálculo de seu hash SHA256. O Resultado do SHA256 é armazenado na sessão do usuário.
- assinar.php → Implementação do handshake OAuth, assim como a utilização dos dois endpoints acima. Como resultado, uma página conforme a figura abaixo será apresentada, mostrando o certificado emitido para o usuário autenticado e a assinatura.

Certificado

-----BEGIN CERTIFICATE----- MIIF/zCCA+egAwIBAgIITZR2ISIYhA0wDQYJK oZIhvcNAQELBQAwgZMxCzAJBgNV BAYTAkJSMREwDwYDVQQKDAhHT1YtQnJ2MTFDMEEGA1UECww6QXV0b3JpZGFkZSBD
ZXJ0aWZpY2Fkb3JhIFJhaXogZG8gR292ZXJubyBGZWRlcmFsIEIyYXNpbDEs MCoGA1UEAwwjQXV0b3JpZGFkZSBDZXJ0aWZpY2Fkb3JhIEhhd2EgR09WQIIwHhcN
MjAwNDIzMTY1NDI3WhcNMjEwNDIzMTY1NDI3WjBLMQswCQYDVQQGEwJCUjELMAkG A1UECAwCU0MxFzAVBgNVBAcMDkZsb3JpYW7Ds3BvbGlzMRYwFAYDVQQDDA1HaW92
YW5pIFBpZXJpMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC60UdD0Vmdod4 l/8Ib/CTT8AApzuQSXbKK7Pnlc9bdJ0mQPpbyJt3LB2RRt2XCpHdrUnbLgDfes
Px+nIN5rPwnRM4H4hp8TJRi3BuodFPt3YgYvltyPUtRdeooBeomIwDXPN2TxUTz4 K9NMbU4Z3PnkjrIYbGg6X/48ycUu5QIDAQABo4ICIDCCAhwwgZUGA1UdEQSBjTCB
iqA4BgVgTAEDAaAvBC0xODA1MTk4NjA1NzUyNjc2OTA4MDAwMDAwMDAwMDAwMDAw MDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwMDAwoB4GBWBMAQMFoBUE
EzAwMDAwMDAwMDAwMDAwMDAwMDCBFWdpb3Zhbmkucc GlcmlAdWZzYy5icjAJBgNV HRMEAjAAMB8GA1UdIwQYMBaAFHFGohdDpp+jFwKhUkXUTi1P417xMEMGA1UdIAQ8
MDowOAYGYEwBAgFjMC4wLAYIKwYBBQUHAgEWIGh0dHA6Ly9yZXBvLml0aS5ici9E UENBQ0hhd2EucGRmMGQGA1UdHwRdMFswLKAqoCiGJmh0dHA6Ly9yZXBvMi5pdGku
YnIvTENSQUNIYXdhZ292YnluY3JsMCugKaAnhiVodHRwOi8vcmVwby5pdGkuYnIv TENSQUNIYXdhZ292YnluY3JsMAsGA1UdDwQEAwIFoDAdBgNVHSUEFjAUBggrBgEF
BQcDBAYIKwYBBQUHAwIwHQYDVR0OBBYEFKbFdXuaYpQUvLIAvB7cVQbIEPWMMGAG CCsGAQUFBwEBBFQwUjAnBggrBgEFBQcwAoYbaHR0cDovLzAuMC4wLjA6ODAwMC9h
aWFtLDdjMCcGCCsGAQUFBzABhhtodHRwOi8vMC4wLjAuMDo4MDAwL2FpYV9wOGMw DQYJK oZIhvcNAQELBQADggIBAIzHO/9kUy4CguK57jV1uCj3hBYnmeOxrBaDso
aAEfkOG4PfrlAyF0hMM91G13RkKNGlP8hz0+YxZb9H5xZ6TmkJPInNmk1sLSl lphB9OSRz0YVzVqL3RJq2lZjm67HMaBvKldn6cC/Y7O6Acs0b4r0snI46EmlZKE
VyKaM5EwIu6nEQkdNrBPoosRKoMLVyAMb6V8WJ226vQZG46GKEhhljdgffeK5fb MLmjtR6pSOLmq47nXdsEYbpuc+MWcfqWMFJz8iUWjQ5SpDdNJSLNXdunbznLMKok
pHstFU0su692JZAon8ZvvDNJckJYj9zNupXENNdzzq/Jngnwkr9AOuPxPR982a S/kM7wRbL6DSQA+IjAsNCfnlO35ToptSXggAyqBbVeO/5V8sZr19cnYy7k8ovtJ WDMq7cF4vAug4JOjf9MnmlYyZ4nmwDLBnq2BlGBj4huTezX/YODjZF+IFH9i2qaq
LcBcryNyQNbOr3epSh4UYDs9mlNDHufjOXYITV8DEVO3nKVgTg52UMFmuT9a10JMp jc1hhLbNrFpFbrDrAqhtp7MYf3J5VEbScHJeXn47rCLVtf6y3bzbb9R4xzahUq
SI9RJK/7TvzPCjRKosI7BYrPcGlGNzZsVHpTdKRdE161jwH+ftNKrKBR+QpAwZE2 4DBn -----END CERTIFICATE-----

Assinatura

EZBBHyj09qWCIKjYeu7EEns1JJ34BkEM211Yl1dymQlF0kLfChIW5CDyrgy3Be8gqL47VL2kugvleGtE9WVOZkJo1jNLz8YxRw7bj5TDMOSwSgK+dgd3IPSHZzdY0eQ3MBowIUBBxty0kwlR0QR82e6WOovlGvo/3/KioeFWZc=

Para executar o exemplo, é possível utilizar Docker com o comando abaixo:

```
docker-compose up -d
```

e acessar o endereço http://127.0.0.1:8080