Assinatura Eletrônica Avançada

Posted on **19/04/2024** by **Cynthia Aurora**

Índice [exibir]

Introdução

A assinatura eletrônica avançada é meio de comprovação da autoria e da integridade de documentos em forma eletrônica.

Histórico

A Medida Provisória nº 2.200-2/2001, Art. 10, § 2o dispõe o seguinte: “O disposto nesta Medida Provisória não obsta a utilização de outro meio de comprovação da autoria e integridade de documentos em forma eletrônica, inclusive os que utilizem certificados não emitidos pela ICP?Brasil, desde que admitido pelas partes como válido ou aceito pela pessoa a quem for oposto o documento.”

Assinar documentos de forma digital deixou de ser inacessível e se tornou algo indispensável para empresas de qualquer porte e setor. A adaptação para o mundo virtual já era uma tendência e ganhou ainda mais força com a pandemia do novo corona vírus.

Em 2020, a MP 983, convertida na lei 14.063, criou uma nova convenção para as assinaturas. Agora, todas elas são chamadas de assinatura eletrônica e estão divididas em três modalidades:

- simples
- avançada
- qualificada

O principal reflexo disso no Brasil é a Lei 14.063, em vigor desde setembro de 2020, que tem como uma das principais novidades a opção da assinatura eletrônica avançada. Esta nova modalidade de assinatura facilitou esse processo e trouxe diversas possibilidades de uso para empresas.

A iniciativa do Governo Federal permitindo a democratização do exercício da cidadania digital no âmbito do poder público, dentro do gov.br, estabelecendo o uso das assinaturas eletrônicas avançadas, baseados no padrão consagrado pela União Europeia (Regulamento nº 914 – eIDAS – vide a norma), e possibilita a adoção ampla e mais segura dos serviços digitais na esfera pública.

Papel do ITI

A partir da publicação da MP 983, o ITI – Instituto Nacional da Tecnologia da Informação ganha novas atribuições. Essas lhe permitirão atuar no âmbito de pesquisas, atividades, normas dos poderes públicos relacionados à criptografia, às assinaturas eletrônicas, à identidade eletrônica e tecnologias correlatas, como blockchain.

Ademais, o ITI proverá uma nova plataforma, segregada da infraestrutura da ICP-Brasil, para o serviço de assinaturas eletrônicas avançadas nas aplicações e canais de acesso do governo, como o gov.br, ampliando sua atuação na esfera pública, proporcionando o acesso dos cidadãos aos componentes seguros (criptográficos), fundamentais para a proteção das transações digitais no âmbito público.

Conceitos

Assinatura eletrônica simples

- Não permite identificar o signatário de forma inequívoca e não utiliza um certificado digital
- Associa dados eletrônicos – como por exemplo a geolocalização – para dar segurança
- Tem menos tecnologias de segurança de dados
- É utilizada em documentos com menos criticidade ou risco, como marcação de consultas médicas e requerimento de informações

Assinatura eletrônica avançada

- Exige tecnologias que assegurem a associação inequívoca dos dados assinados com a identidade
- Pode utilizar o certificado corporativo avançado, que utiliza as mesmas tecnologias de verificação de proteção de dados que a ICP-Brasil, e biometria

- Tem mais abrangências que a simples e menos que a qualificada
- Indicada para documentos que envolvem informações de maior criticidade ou risco: contratos corporativos, alteração e encerramento de empresas, acesso a documentos e atualização de cadastros

Assinatura eletrônica qualificada

- Deve usar certificado digital reconhecido pela Infraestrutura de Chaves Públicas Brasileira, a ICP-Brasil
- Modalidade já era permitida e regulamentada desde 2001, pela MP 2.200
- É válida para qualquer documento ou transação
- Indicada para documentos que tenham alto grau de criticidade ou risco
- Obrigatória para: atos assinados por chefes de poder, interações que envolvam sigilo constitucional, legal ou fiscal; notas fiscais eletrônicas e transferências de bens ou imóveis
- Essa nova classificação foi criada para simplificar e facilitar a verificação de assinaturas, mas é importante ressaltar que essa lei regulamenta as assinaturas digitais do setor público, e desde então vem sendo utilizada como norte também no setor privado.

Assinatura Eletrônica Avançada – gov.br

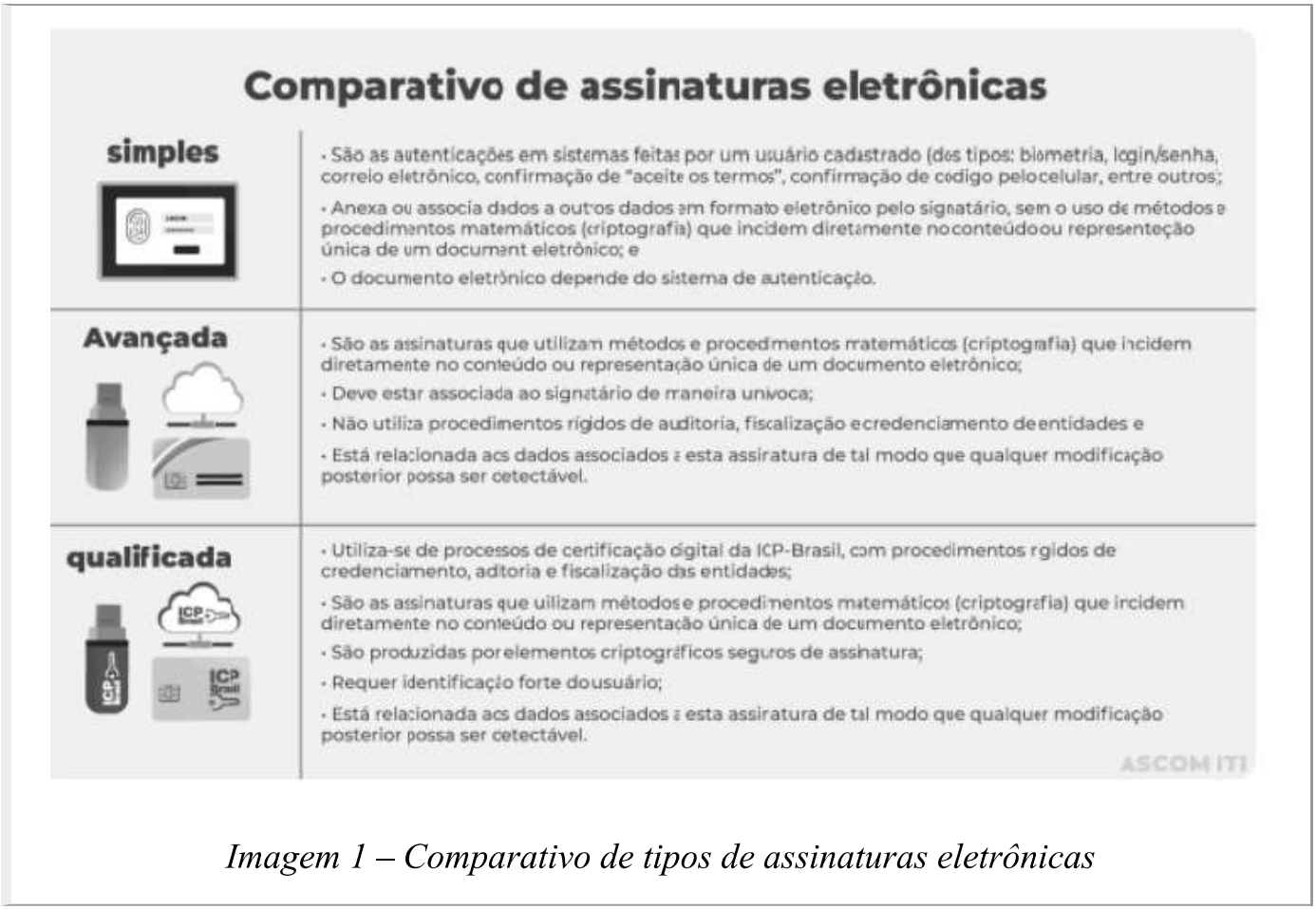

Imagem 1 – Comparativo de tipos de assinaturas eletrônicas

Para atender a demanda de processamento de Assinatura Eletrônica Avançada de documentos eletrônicos e com o objetivo de agilizar, reduzir custos, desburocratizar e prover segurança de autenticidade e integridade e não-repúdio a Procergs pesquisou várias alternativas de solução e escolheu o provedor de assinaturas do gov.br, pois este possui um alto nível de segurança na identificação do usuário e além disso, possui o aporte tecnológico aderente aos requisitos estabelecidos pelas aplicações providas pela Procergs à Administração Pública Estadual.

VANTAGENS

- A agilização dos processos administrativo com o cidadão, com automatização de processos.
- Redução de custos para o cidadão.
- Redução de custos da Administração Pública Estadual.
- Maior segurança em procedimentos realizados pela internet pelo cidadão.

- Pode ser utilizado em todas as plataformas Windows, Linux ou Mac.
- Não há necessidade de configuração prévia na estação de trabalho.
- A solução pode ser utilizada em dispositivo móvel (Android ou iOS) para assinatura de documentos e autenticação.
- A segurança e controle das transações se dará através do PSC-Provedor de Serviço de Certificação Gov.Br
- Não há necessidade da instalação de aplicação no dispositivo no qual será realizada a assinatura com o certificado do Gov.br.
- O procedimento de emissão do certificado é automático. Diferente do processo de emissão de certificados pela cadeia da ICP-Brasil.

Requisitos do gov.br

REQUISITOS PARA O USUÁRIO

- Ter certificado digital gov.br e deve possuir nível prata ou ouro de identificação.
- Ter um computador ou dispositivo móvel, celular, pode ser Android ou IOS.
- Acesso a internet.
- O usuário deve entender os processos para que possa realizar o cadastramento e inclusão nos níveis no gov.br.

REQUISTOS PARA INTEGRAR A APLICAÇÃO

- Para que a aplicação utilize o processo de assinatura eletrônica avançada é necessário que a aplicação realize o procedimento de Login – autenticação no gov.br conforme a definição que consta no manual de integração.
- É premissa de toda aplicação, que irá consumir os serviços da API de assinatura avançada, estar integrada à plataforma de autenticação digital do Cidadão – Login Único.
- O Provedor gov.br solicita a informação do órgão ou entidade consumidora das APIs de assinaturas eletrônicas avançadas e **da descrição do serviço, da volumetria anual estimada da quantidade de documentos que serão assinados e da sazonalidade da efetividade de assinaturas.**

Requisitos integração Procergs

- A aplicação deve ter integração com o serviço de assinatura eletrônica avançada pelos componentes fornecidos, tanto para a plataforma Java como .Net, e estar aderente as regras estabelecidas.
- O uso de assinatura eletrônica avançada deve ser realizado em procedimentos e atendimentos que envolvam o cidadão. **Não deve ser utilizado para processo administrativos internos** da Administração Pública Estadual. Esta orientação é oriunda do governo federal.
- O Provedor gov.br solicita a informação do órgão ou entidade consumidora das APIs de assinaturas eletrônicas avançadas e **da descrição do serviço, da volumetria anual estimada da quantidade de documentos que serão assinados, da sazonalidade da efetividade de assinaturas.** Esta informação deve ser passada por meio de uma **REQ** requisição em nossa plataforma de atendimento de serviço. **https://atendimento.procergs.rs.gov.br/**, por meio da requisição **Outros.Solicitar.assinatura eletrônica** GOV.BR, que será o responsável por reunir e repassar as informações ao provedor gov.br .
- A **REQ – Requisição de Serviço** deve ser feita quando for iniciado o desenvolvimento da aplicação. As configurações necessárias para que a aplicação se integre ao componente serão liberadas, somente, após à aprovação da REQ.
- A Procergs informará ao provedor Gov.Br. baseada nas quantidades estimadas por cada uma das aplicações usuárias da assinatura eletrônica avançada. A informação deve ser enviada antecipadamente, sempre que tiver informação prévia, acerca de aumento representativo da demanda informada quando da habilitação inicial, sob pena de **ter o acesso desabilitado** para não prejudicar as demais aplicações habilitadas.
- O provedor govbr pede que quando houver previsão de uma demanda maior que esta seja informada. **Informar o período de aumento da demanda, caso possa ocorrer**.
- A aplicação deve trabalhar de acordo com o Decreto Nº 56671 DE 26/09/2022 – Rio Grande do Sul, que regulamenta os tipos de assinaturas que têm efeito no meio eletrônico.

Arquitetura da Assinatura Eletrônica Avançada

A arquitetura provida para o uso assinatura eletrônica avançada segue a mesma estrutura utilizada nos procedimentos de assinatura com certificados digitais. Desta forma, as aplicações continuarão utilizando os componentes de apoio, `ACRSUTILARQ` ou `ACRSUTILARQFLAT`, para o preparo do documento a ser assinado. O componente de apoio `ACRSUTILARQ` deverá ser utilizado para a geração do *hash* que será enviado para a assinatura e no retorno do *hash* assinado para a verificação e visualização do documento assinado.

Fluxo do Oauth 2.0

O acesso aos certificados digitais do Gov.br é semelhante ao acesso aos certificados em nuvem. A diferença está que no primeiro acesso há a geração do certificado, se o usuário tiver nível prata ou ouro na identificação perante ao registro do Gov.br

Há o uso do protocolo Oauth2.0, conforme o regulamentado na instrução Normativa Nº 6 do ITI de 16 de abril de 2018. A integração com todoas provedores de certificado para a assinatura eletrônica deverá seguir o mesmo fluxo, que é estabelecido pelo ITI. Os padrões definem como as aplicações devem se integrar com a plataforma Gov.br ou outra que venha surgir, para que o acesso aos certificados para assinatura eletrônica avançada se dê da mesma forma. Os certificados digitais do Gov.br ficam armazenados em um placa criptográfica, HSM-Hardware Security Module de um Provedor.

O detalhamento deste fluxo tem como objetivo demonstrar como é realizado o processo de acesso ao certificado pela infraestrutura desenvolvida para apoiar a integração com as aplicações da Procergs.

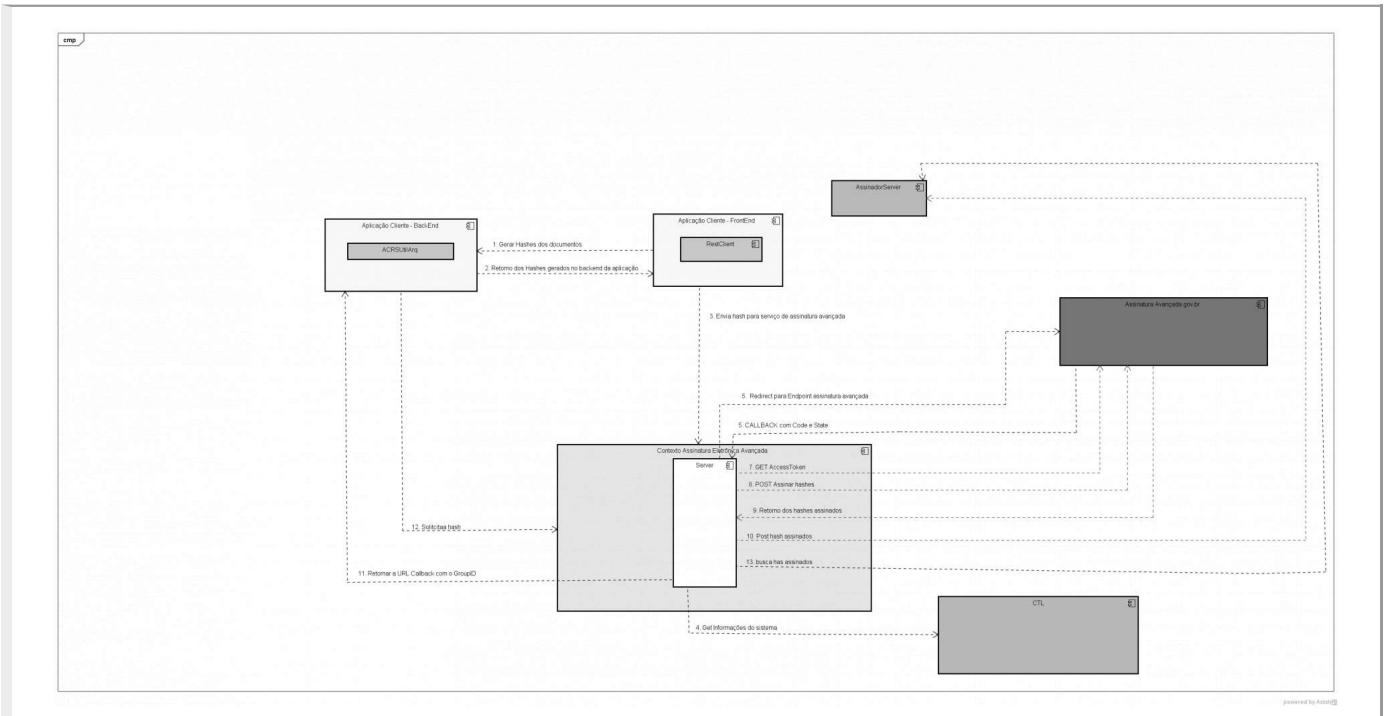

Imagem 2 – Diagrama do fluxo da assinatura Eletrônica Avançada

Segue o fluxo do *OAuth* 2.0 , que aplicação cliente da infraestrutura da Procergs realiza para o uso do certificado de assinatura eletrônica avançada. Estão descritas as interações entre as partes:

1. A aplicação cliente solicita para o contexto de certificação digital a realização de uma assinatura eletrônica.
2. O contexto de certificação digital redireciona para a plataforma Gov.br.
3. A plataforma Gov.br fica com o controle do processo. No primeiro acesso do usuário, se ele tiver o nível prata ou ouro, um certificado digital é emitido para o cidadão.
4. Caso o cidadão tenha o nível bronze a plataforma Gov.br retornará uma exceção repassando a seguinte mensagem Http Code: 403 Erro: Cidadão não possui a identidade (Prata ou Ouro) necessária para uso da assinatura eletrônica digital.
5. Com o certificado na mão há um pedido de autorização feito diretamente para o proprietário do certificado para seu uso com envio de um código para o cidadão autorizar o uso do certificado.
6. Após a autorização na plataforma Gov.br, o contexto de certificação digital recebe uma concessão de autorização (`CODE`), que é uma credencial representando a autorização do proprietário do certificado.
7. O contexto de certificação digital solicita um *token* de acesso apresentando o código de autorização(`CODE`);
8. O servidor de autorização autentica o cliente e valida a concessão de autorização e, se for válido, emite um *token* de acesso;
9. O contexto de certificação digital utiliza esse *token* para realizar uma (ou várias) assinatura(s).

Como iniciar processo de assinatura avançada

CADASTRO DAS APLICAÇÕES

Para que as aplicações utilizem o serviço de Assinatura Eletrônica Avançada, deve ser realizado um cadastro prévio. Para isso, deve-se enviar uma solicitação de cadastro para o e-mail certificacaodigital@procergs.rs.gov.br com as seguintes informações:

- **Sigla do Sistema**
- **Chave –** Se a aplicação já utiliza o Assinatura Digital, passar a chave atual, caso contrário, será gerado uma nova chave.

- **Urls de callback** – URLs de *callback* que serão utilizadas para finalizar o processo de assinatura.

REQUISITOS DE VERSÕES DE COMPONENTES

- **Utilize a última versão disponível do ACRSUTILARQ.**

Para uso da tecnologia de assinatura eletrônica avançada foi elaborada uma arquitetura semelhante a arquitetura de certificação em nuvem pois, a infraestrutura do provedor Gov.br é semelhante a infraestrutura e os processos necessário para integração da aplicação cliente com a aplicação de infraestrutura de backend Procergs é similar.

Arquitetura para as aplicações

O fluxo para as aplicações efetuarem uma assinatura está descrito na imagem abaixo:

1. A aplicação cliente irá realizar um POST para o nosso contexto de certificação, passando todos dados referente a uma assinatura eletrônica avançada.
2. O contexto de certificação irá processar os dados e gerar uma URL para a interface do assinadorservices para as aplicações realizarem o *redirect*, caso a aplicação cliente utilize algum tipo de assinatura além da Avançada;
3. Quando todo fluxo do OAuth2.0 entre o provedor Gov.Br e o contexto de certificação estiver concluído, o contexto de certificação irá chamar a URL de *callback* da aplicação cliente com o groupID (code) da transação;
4. A aplicação cliente, usando esse groupID (code), efetuará um GET para buscar os *hashes* dos arquivos assinados que estarão armazenados no contexto de certificação digital da Procergs;
5. De posse dos *hashes* assinados, o fluxo de verificação da assinatura e armazenamento seguem sendo os mesmos já utilizados nas aplicações atualmente.

OBTENDO TOKEN DA APLICAÇÃO

Aplicações que já utilizam o AssinadorFX devem utilizar a mesma chave que foi fornecida pela equipe de Certificação Digital. Para as aplicações que vão utilizar pela primeira vez a tecnologias de assinatura eletrônicas, enviar um e-mail para certificacaodigital@procergs.rs.gov.br solicitando a chave.

Obs.: Além do token, é feito o cadastro para o uso da assinatura eletrônica avançada, este só é realizado após o envio da REQ por meio de nossa plataforma de serviços USD **https://atendimento.procergs.rs.gov.br/**, requisição **Outros.Solicitar.assinatura eletrônica** GOV.BR. Somente após este cadastro é possível realizar os acessos para integração com a plataforma Gov.Br.

GERANDO HASH DE UM ARQUIVO

Abaixo um exemplo de como gerar o hash de um arquivo que será enviado no JSON. **O Hash do arquivo deve ser gerado no servidor**.

```
import com.acrs.utilarq.UtilArqExtrai;
import com.acrs.utilarq.UtilArq;
public void geraHash() throws UnsupportedEncodingException{



 byte[] arrayDocumento = null; // Buscar array de bytes do arquivo a ser assinado
           String nomeArquivo = ""; //Nome do arquivo
           UtilArqExtrai utilArqExtrai = new UtilArqExtrai(arrayDocumento, nomeArquivo);
           String resumo = utilArqExtrai.getXMLResumo();
           String hash = UtilArq.codificaByteParaString(resumo.getBytes("UTF-8"));     }
```

MONTANDO JSON PARA ENVIO

Deve ser enviado o seguinte JSON para o contexto de certificação:

- **documento –** Informar CPF para assinaturas dos usuários do Gov.br;
- **sistema (obrigatório)** – Sigla do sistema
- **urlCallback (obrigatório)** – URL de callback da aplicação. Após o processo de assinatura, a aplicação de Certificação irá chamar essa URL passando o atributo groupID;
- **escopo (obrigatório)** – Informar os valores ASSINATURA

```
{
    "documento": "",
    "urlCallback": "http://localhost:4200/assets/retorno.html",
    "restricao": "AVANCADA"
    "iss": "TST" // sistema emissor do token
    "iat": 1662482919116 // long que representa a data/hora de geração do token, pode ser obtido com
    "arquivos": [
       {
      "fileId": "1",
      "docName": "DECRETO Nº 10.543, DE 13 DE NOVEMBRO DE...MBRO DE 2020 - DOU - Imprensa Nacional.p
      "fileHash": "PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0iVVRGLTgiPz4NCjxoYXNoPg0KICA8U0hBMjU2IG5vb
    }
    ]
```

DEPENDÊNCIAS NECESSÁRIAS

Maven

```
   <!-- Dependência de certifição para auxiliar durante o processo -->
        <dependency>
            <groupId>com.acrs</groupId>
            <artifactId>ACRSUtilArq</artifactId>
            <version>1.6.11</version>
        </dependency>
    <!-- JWT -->
    <!-- Sugestão: Dependência para gerar o JWT  -->
        <dependency>
            <groupId>com.nimbusds</groupId>
            <artifactId>nimbus-jose-jwt</artifactId>
            <version>9.11.1</version>
        </dependency>
```

Obs.: Verificando incompatibilidade de dependências do componente ACRSUTILARQ e aplicação cliente, envie e-mail para certificacaodigital@procergs.rs.gov.br solicitando apoio para solucionar a questão. Um exemplo de dependência que pode gerar incompatibilidade é a library do ITEXT.

GERAR JWT

Para enviar o JSON gerado para o contexto de certificação?? cerservices (assinadoservices)?? é necessário assiná-lo. Segue abaixo um exemplo de código para assinar o JSON e outro para verificar a assinatura.

```

public String assinarJWT( String chaveAplicacao, String json) {
    try {
String jwtKey =  UtilCrypto.getCryptoToken(chaveAplicacao);
      log.debug("Assinando JWT. Secret: {} Json: {}" ,jwtKey, json);
      // Criando objeto JWSSigner com algoritmo HMAC256
      JWSSigner signer = new MACSigner(jwtKey);
      // Preparando objeto JWS com json de payload
      JWSObject jwsObject = new JWSObject(new JWSHeader(JWSAlgorithm.HS256), new Payload(json));
      // Assinando objeto JWS
      jwsObject.sign(signer);
      return jwsObject.serialize();
    } catch (JOSEException e) {
      throw new GenericException("Erro ao assinar JWS com HMAC256. Erro: " + e.getMessage(), e);
    }
  }
```

EXEMPLO JWT

```
eyJhbGciOiJIUzI1NiJ9.CQkJewogICAgImRvY3VtZW50byI6ICI3NTI3NTE1NTAzNCIsCiAgICAic2lzdGVtYSI6ICJSREEiLAo
YWNrIjogImh0dHA6Ly9sb2NhbGhvc3Q6NDIwMC9hc3NldHMvcmV0b3Juby5odG1sIiwKICAgICJlc2NvcG8iOiAiQVNTU5BVFVS
HJpbWFyaWEiOiAiYmx1ZSIsCiAgICAiY29yU2VjdW5kYXJpYSI6ICJncmV5IiwKICAgICJhcnF1aXZvcyI6IFsKICAgICAgICB7C
JmaWxlSWQiOiAiMSIsCiAgICAgICAgICAgICJkb2NOYW1lIjogIkNvbnZlcnRpZG9HQS5wZGYiLAogICAgICAgICAgICAiZmlsZU
Xd2dkbVZ5YzJsdmJqMGlNUzR3SWlCbGJtTnZaR2x1WnowaVZWUkdMVGdpSUhOMFlXNWtZV3h2Ym1VOUlubGxjeUkvUGdvOGFHRnp
alUySUc1dmJXVkJiR2R2Y21sMGJXODlJakl1TVRZdU9EUXdMakV1TVRBeExqTXVOQzR5TGpFaVBsSjJkbmhLUW5JMGJ6VmtjWGRE
XpkQkwyZ3pVVGM0Y1dSNVlXaHVWREp5TDJzOVBDOVRTRUV5TlRZK0NpQWdQRk5JUVRFZ2JtOXRaVUZzWjI5eWFYUnRiejBpTVM0e
5pSStaMnN3T1V4WUwwVldkbXBWSzNZelJHTjZkRlE0SzBsblYxcFJQVHd2VTBoQk1UNEtJQ0E4VTBoQk1qSTBJRzV2YldWQmJHZH
NVFl1T0RRd0xqRXVNVEF4TGpNdU5DNHlMalFpUGp3dlUwaEJNakkwUGdvZ0lEeFRTRUV6T0RRZ2JtOXRaVUZzWjI5eWFYUnRiejB
TVM0eE1ERXVNeTQwTGpJdU1pSStUbVJFZG14b2NGZENhSE4xY0hkemFVUkViVGRpVjBGMlVrOTRWRFIxUVhKNFJXTlhWMmRzTlV0
VpXWlJWVkpWYTBOR1dETnhUand2VTBoQk16ZzBQZ29nSUR4VFNFRTFNVElnYm05dFpVRnNaMjl5YVhSdGJ6MGlNaTR4Tmk0NE5EQ
BMakl1TXlJK1Mwd3ZOMHR2VFM4emVWQnBTbFJxYkRFckwyaDBaemxhUnpSeFlXMDNabFZFZFhCNVJFWnlaM1pRYm14b1dtUTBkRF
2VFVsYU9HeG5OWE5JV2xsTlNDdDBkRFF2Um05d2RXcGhSbEU5UFR3dlUwaEJOVEV5UGdvZ0lEeE5SRFVnYm05dFpVRnNaMjl5YVh
ME1DNHhNVE0xTkRrdU1pNDFJajVCVHpONVkwTlFjazgxYlVGUWFEaHBjVTUyTVZWQlBUMDhMMDFFTlQ0S1BDOW9ZWE5vUGc9PSIs
CJtaW1lVHlwZSI6ICJhcHBsaWNhdGlvbi9wZGYiLAogICAgICAgICAgICAiZmlsZU5hbWUiOiAiQ29udmVydGlkb0dBLnBkZiIKI
BdCn0.IvG4lQnOLhZN3683RfqwM9yMTZtjvzH8D06RKrARjbs
```

GERAR PAR DE CHAVES DE AUTENTICAÇÃO

Para simplificar o desenvolvimento, foram introduzidos métodos no componente UtilArqFlat a fim da geração do par de chaves, que podem ser utilizadas conforme os exemplos abaixo.

As requisições no contexto de certificação avançada requerem que as aplicações utilizem um par de chaves nos campos code_challenge e code_verifier como reforço de segurança. Essa medida é uma recomendação da RFC 7636. Para saber mais sobre o PKCE, no item Referências tem o endereço do link da explicação e da RFC.

```
String[] parChaves = UtilCrypto.getChavesAutorizacao(); // ou UtilCrypto.getChavesAutorizacao("VALOR
String code_challenge = parChaves[0];
String code_verifier = parChaves[1]; // armazenar o valor desse campo para busca dos hashes;
```

Caso a aplicação já possua algum valor aleatório que possa ser trafegado de forma segura em URLs, pode passá-lo para o método getChavesAutorizacao() para obter o valor a que deve ser enviado na chamada POST /assina.

CHAMADA POST /ASSINA

Realizar um `POST` para:

Homologação: https://sechml.procergs.com.br/assinadorservices/rest/assina?code_challenge=<CHAVE GERADA>

Produção: https://secweb.procergs.com.br/assinadorservices/rest/assina?code_challenge=<CHAVE GERADA>

Header

- `Content-Type` : (string) application/json

Body:

Informar o JWT gerado.

Resultado da solicitação

SOLICITAÇÃO VÁLIDA (SUCESSO): 200

Se a solicitação estiver válida, o contexto de assinatura irá retornar a lista de hashes assinados.

- `fileId` – Irá retornar o ID do arquivo que foi informado no JSON enviado no inicio do processo de Assinatura.
- `fileHashSigned` – Irá retornar o hash assinado do arquivo

Exemplo de JSON:

```
JsonReader jsonReaderMensagem = Json.createReader(new StringReader(retorno));
                    uriRetorno = jsonReaderMensagem.readObject();
                    uriRedirect = uriRetorno.getString("uri");
```

```
{"uri":"https://cas.staging.iti.br/oauth2.0/authorize?response_type=code&scope=sign&client_id=devLoca
```

COMO BUSCAR OS HASHES ASSINADOS

Após o processo de assinatura finalizar, o contexto de assinatura irá retornar na URL de callback passada no POST para iniciar o processo, um novo parâmetro, `code.`

Exemplo da url de Callback:

```
https://secweb.des.intra.rs.gov.br/rda/mod-tarefa/tarefadocumento-form.xhtml?
code=9005&idTarefaInstancia=2101525&groupId=942e8e2e8958b151be32834a6efeb0115da9cecc71f0c2ca822bcbcf5
036c375973c4d397ea12a6d3fcf3cf324906a0011ed72f9d7a8c85a
```

Recebendo este `code` , as aplicações devem buscar os hashes assinados, conforme descrito abaixo, e finalizar o processo de assinatura.

CHAMADA GET /ASSINADORSERVICES/HASHES/<CODE>

Depois de obter o retordo no callback, a aplicação deve buscar os hashes assinados, para isso, ela deve fazer uma chamada GET para o endpoint /assinadorservices/hashes/ informando o `code` recebido como parâmetro na query. Além disso, deve encaminhar o `code_verifier` gerado na primeira requisição.

Realizar um `GET` para:

- **Homologação:** https://sechml.procergs.com.br/assinadorservices/rest/hashes/<code>
- **Produção:** https://secweb.procergs.com.br/assinadorservices/rest/hashes/<code>

```
curl --location --
request GET "https://sechml.procergs.com.br/assinadorservice/rest/hashes/<code>942e8e2e8958b151be3283
c2ca822bcbcf54da64f4676eef3a3036c375973c4d397ea12a6d3fcf3cf324906a0011ed72f9d7a8c85a?code_verifier=</
vN3q8DCJyy8rX_NZ41DnPaBHje0KLHsJgMnE<code>" \
--data-raw ""
```

Resultado da solicitação

SOLICITAÇÃO VÁLIDA (SUCESSO): 200

Se a solicitação estiver válida, o contexto de certificação (cerservice) retornará a lista de hashes assinados.

- `fileId` – Irá retornar o ID do arquivo que foi informado no JSON enviado no inicio do processo de Assinatura.
- `fileHashSigned` – Irá retornar o hash assinado do arquivo

Exemplo de JSON:

```
[
    {
       "fileId":"1",
       "fileHashSigned":"MIAGCSqGSIb3DQEHAqCAMIACAQExDzANBglghkgBZQMEAgEFADCABgkqhkiG9w0BBwEAAKCAMIIG
    }
 ]
```

Exemplo processamento dos arquivos assinados recebidos

```
// carrega o arquivo original
File arquivoOriginal = new File("....");
// itera sobre o retorno do contexto de certificação
JsonReader jsonReaderMensagem = Json.createReader(new StringReader(retorno));
                 JsonArray arquivos = jsonRetorno.readArray();
                         for (int i = 0; i < arquivos.size(); i++) {

                // para cada arquivo assinado utiliza-se um arquivo temporário
        String signedFileHash = arquivos.getJsonObject(i).getString("fileHashSigned");
                File arquivoTemporario = new File("....");

                // decodifica a assinatura para um array de bytes
                UtilIO.gravaArquivo(UtilArq.decodificaStringParaByte(signedFileHash), arquivoTempora

                // empacota o arquivo original e o arquivo temporário a fim de extrair o PKCS7 das a
                UtilEmpacota utilEmpacota = new UtilEmpacota(arquivoOriginal, arquivoTemporario);
        byte[] pkcs7Empacotado = utilEmpacota.getPKCS7();
        UtilArqExtrai arq = new UtilArqExtrai(new ByteArrayInputStream(pkcs7Empacotado));

                // decodifica o array de bytes para string para verificação da assinatura
                String pkcs7 = UtilArq.codificaByteParaString(pkcs7Empacotado);
        String resumo = arq.getXMLResumo();
        Stringxml = new PRAssinadorWSClient(newURL(prassinadorws))
                    .verificaPKCS7Sistema(resumo,pkcs7,  sistema, EscopoAssinatura.AVANCADO);
        String chave = "...";
        String url_visualizacao = "....";

        String XMLCRC = UtilPDF.getXMLVisualizacaoCRC(chave, url_visualizacao);
        InputStream inputStream = arq.getDocumentoVisualizacao(xml, true, XMLCRC);
             ... destina arquivo de visualização
```

Exemplo de visualização do arquivo assinado

```
// Verifique na documentação no componente ACRSUTILARQ qual o método de visualização mais adequado pa
UtilArqExtrai utilArqExtrai = new UtilArqExtrai( );
Inputstream docFormatado = utilArqExtrai.getDocumentoVisualizacao(
                          <String xmlAssinatura> <boolean flagRodapPaginacao>
                          <String chave>, <int numeroPagina>
                          <String qrCodeMsg>,<String endereco>);
```

Solicitação inválida (Erro)

Se a solicitação for considerada inválida, o contexto de assinatura irá retornar um **JSON** contendo a mensagem referente ao problema e um **status code de erro**.

IMPORTANTE: Qualquer erro que for registrado durante o processo de assinatura será retornado nesta consulta.

Principais erros retornados:

```
Status: 400 Bad Request
{
    "mensagem":"Erro ao iniciar processo de assinatura.",
    "mensagemDetalhada":"As URLs de callback informadas na requisição são desconhecidas ou não estão
 }
```

```
Status: 401 Not Authorization
{
    "mensagem":"token inválido.",
    "mensagemDetalhada":"Erro ao iniciar processo de assinatura."
 }
```

```
Status: 403 Access denied
{
    "mensagem":"Acesso negado!",
    "mensagemDetalhada":"Cidadão não possui a identidade (Prata ou Ouro) necessária para uso da assin

}
```

```
Status: 408 Expired Session
{
    "mensagem":"Sessão está expirada!",
    "mensagemDetalhada":"Sessão está expirada, refaça o acesso."
 }
```

```
Status: 500 Error not expected ???
{
    "mensagem":"Erro inesperado",
    "mensagemDetalhada":"Ocorreu um erro inesperado com a aplicação do Assinador Service"
 }
```

```
Status: 403 Not Found
{
    "mensagem":"Não foi encontrado processo de assinatura pelo id informado!",
    "mensagemDetalhada":"Não foi encontrado processo de assinatura pelo id informado!"
 }
```

```
Status: 404 Not Found
{
    "mensagem":"access_denied",
    "mensagemDetalhada":"Operação cancelada pelo usuário!"
 }
```

Referências

- https://manual-integracao-assinatura-eletronica.servicos.gov.br/en/latest/iniciarintegracao.html#orientacoes-para-testes-em-ambiente-de-homologacao
- https://manual-roteiro-integracao-login-unico.servicos.gov.br/pt/stable/iniciarintegracao.html#resultado-esperado-do-acesso-ao-

servico-de-confiabilidade-cadastral-selos
- https://www.gov.br/iti/pt-br/central-de-conteudo/parecer-agu-pdf
- https://www.gov.br/iti/pt-br/central-de-conteudo/parecer-pdf/@@download/file/parecer.pdf
- http://www.planalto.gov.br/ccivil_03/_Ato2019-2022/2020/Decreto/D10543.htm
- https://www.legisweb.com.br/legislacao/?id=409786
- https://www.congressonacional.leg.br/materias/medidas-provisorias/-/mpv/142535
- https://erikshimoda.medium.com/aumentando-a-seguran%C3%A7a-com-pkce-8b2c8c2e4b64
- https://www.rfc-editor.org/rfc/rfc7636