MARÇO DE 2023

MANUAL DO DESENVOLVEDOR

COMO UTILIZAR AS OPEN APIS DO BRADESCO

API STUDIO
BANCO BRADESCO
https://developers.bradesco.com.br/

ÍNDICE

OBJETIVO

O presente manual demonstra como as **Open APIs** do Banco Bradesco devem ser utilizadas. Nele você verá como o processo de **autenticação** e **autorização** deve ser realizado, quais os passos para criar cada um dos artefatos necessários para consumo das APIs, além de um glossário com os termos técnicos presentes no documento.

AUTENTICAÇÃO

O acesso às **Open APIs** do Bradesco é feito através do uso alguns protocolos e padrões técnicos para garantir a segurança no tráfego das informações. Sempre que for necessário consumir alguma API do Bradesco, é necessário realizar o processo de autenticação.

Nesse processo, o sistema que quer se conectar em nossas APIs precisa ser previamente conhecido pelo Bradesco. Para isso é necessário seguir os passos abaixo:

1. Criação de um Client ID

Para solicitar a criação de um Client ID, é necessário enviar um e-mail para suporte.api@bradesco.com.br seguindo o *template* "**Cadastro de *Client ID*"**.

Como toda comunicação de APIs do Bradesco com parceiros é feita utilizando o **protocolo mTLS** é necessário o uso de um **certificado digital**. Existem algumas regras específicas para os certificados digitais aceitos pelo Bradesco. Confira abaixo:

- Deve seguir o **padrão ICP-BRASIL** do tipo **A1**
- **Tamanho mínimo** de 2048 bits
- Utilizar algum **algoritmo** RSA como o RSASHA 256, 384 ou 512
- **Data de validade/expiração** deve ser superior há 4 meses e no máximo 3 anos
    - Data de validade a contar da **data da solicitação** de cadastramento ou renovação

Para fins de teste, o certificado usado no ambiente de homologação pode ser uma versão auto assinada. Se você precisar de ajuda para gerar essa versão de teste, leia o guia de referência **"Criando um certificado auto assinado".**

Devido ao uso do protocolo mTLS, é necessário que a comunicação seja feita usando um dos algoritmos/cifras abaixo:

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, e/ou

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

Se sua solicitação estiver dentro dos requisitos definidos pelo banco, em até 3 dias forneceremos um *Client ID* ou *Client Key* de acesso com informações de *URL* do ambiente e URL de geração de token (autorização).

Com o *Client ID* / *Client Key* será possível sua aplicação provar que possui autenticidade para usar nossas APIs. O próximo passo é garantir que existe autoridade para usar determinada API.

AUTORIZAÇÃO

Assim como especificado pelo padrão **OAuth 2.0**, sempre que você quer consumir uma API é necessário provar que sua aplicação tem autorização e autenticidade para usá-la.

Para provar que existe autorização para acessar uma API, é necessário obter um **token JWT** através do nosso serviço de autorização**.** Esse *token* traz diversas informações sobre sua aplicação codificadas em **Base64** e serão usadas nas chamadas feitas para a API autorizada.

Para pedir um *token* JWT, siga o procedimento "**Gerando token de acesso em homologação (*access token*)".** Com um token em mãos, o próximo passo é o de consumo da API de fato.

CONSUMO

1. Consumo em homologação

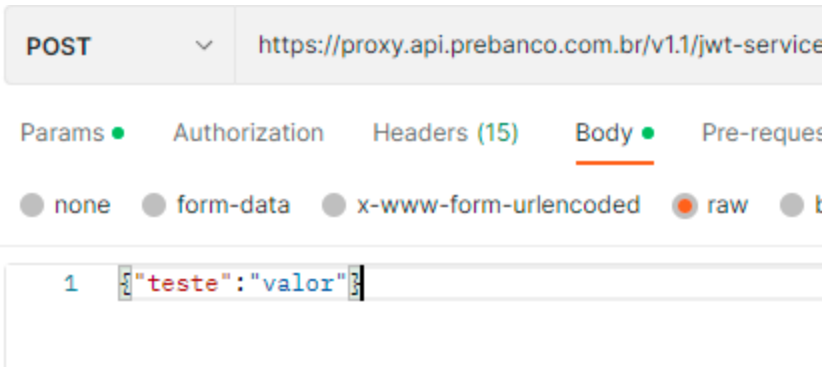
Assim como é feita a assinatura de um JWT para envio durante o momento de autorização, no momento de consumir APIs é necessário criar uma nova assinatura e enviá-la no ***header* da requisição**.

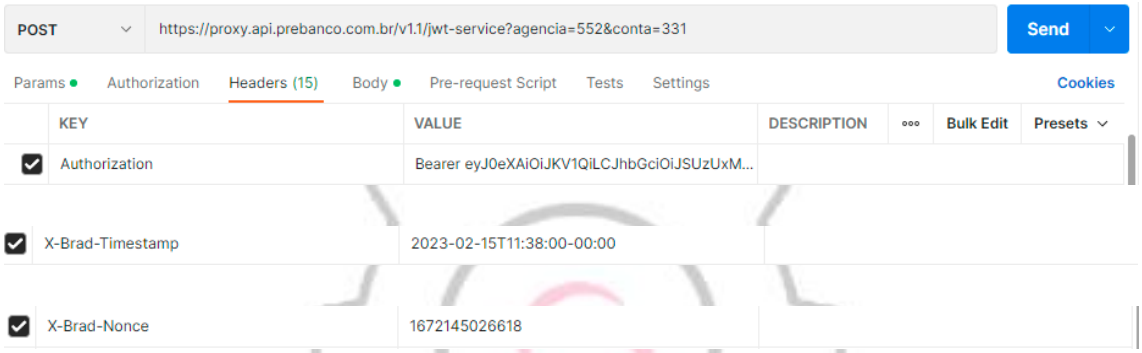
Para saber como essa assinatura deve ser montada, veja o guia de referência **"Criando o header *X-Brad-Signature* para consumo de APIs".** Além disso, deve ser enviado como header o campo ***"X-Brad-Nonce"***, que deverá ser um valor aleatório usado a cada chamada. Podemos usar, por exemplo, a data atual em milissegundos (a data atual em milissegundos, 1672145026618).

Também é necessário passar o parâmetro "***X-BradTimestamp"*** no header, contendo o *timestamp* de quando o *endpoint* está sendo chamado. Necessário seguir essas regras:

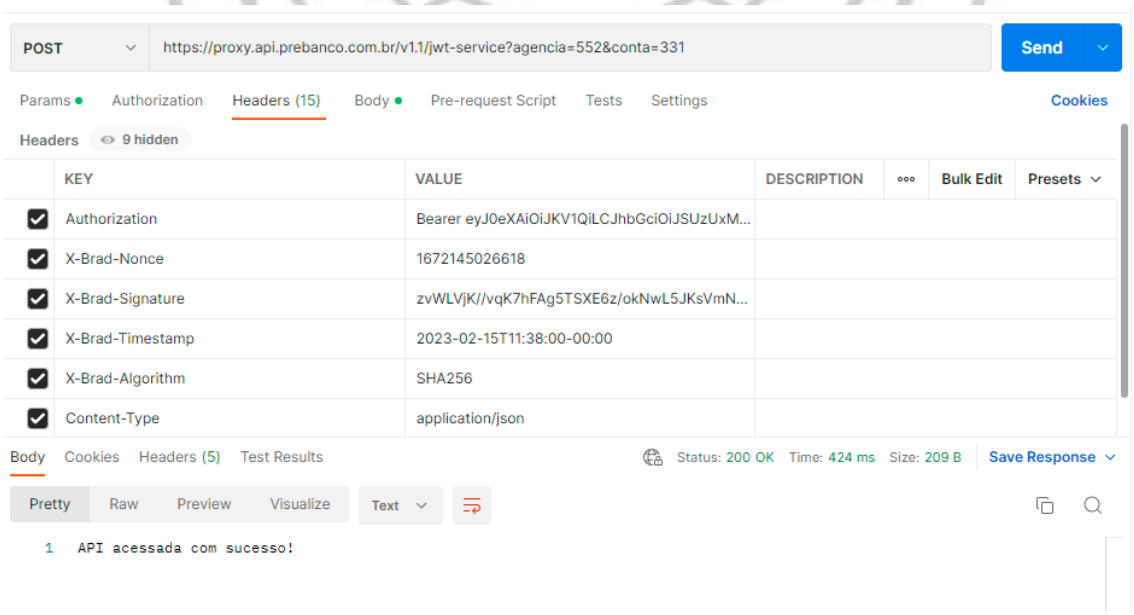
- Formato "**AAAA-MM-DDThh:mm:ss-00:00**", sendo:
    - **AAAA** = ano com quatro caracteres; **exemplo** "2023"
    - **MM** = mês com dois caracteres; **exemplo** "02" referindo-se ao mês de fevereiro;
    - **DD** = dia com dois caracteres; **exemplo** "15"
    - **T** = parâmetro fixo;
    - **hh** = hora com dois caracteres; **exemplo** "11", referindo-se às 11 da manhã
    - **mm** = minutos com dois caracteres; **exemplo** "38"
    - **ss** = segundos com dois caracteres, **exemplo** "00";
    - -00:00 = diferença para o fuso horário UTC 0:00.
        - Pode ser utilizado UTC -3:00 (fuso horário Brasília)

Dessa forma a requisição ficará assim:





Para validar se todos os passos foram feitos corretamente, você pode testá-los usando esse *endpoint*:



Exemplo de uma requisição de teste bem-sucedida

2. Consumo em Produção

Após ter concluído todos os testes com sucesso em ambiente de homologação, o próximo passo é solicitar as credenciais do ambiente de produção. Para isso, será

necessário adquirir um certificado emitido por Autoridade certificadora, como: *Digicert*, *CertiSign*, Serasa entre outras. Geralmente o certificado emitido pelas Autoridades Certificadoras é disponibilizado em um arquivo formato ".pfx".

Este arquivo contêm uma "chave pública e uma "chave privada", ao qual a chave-pública deverá ser extraída do arquivo ".pfx" para envio ao Bradesco. Solicite a empresa emissora do certificado para que lhe forneça os arquivos de chave pública e privada do mesmo.

O Certificado SSL deve ter **padrão ICP-Brasil e ser do Tipo A1** seguindo as características mencionadas anteriormente. Também recomendamos que no momento da aquisição do certificado SSL seja solicitado à empresa que irá gerar o novo certificado que forneça os arquivos de **chave pública** e **privada** no formato ".pem" e ".cer", facilitando posteriormente o envio do arquivo ".pem" ou ".cer" referente a chave pública para o Bradesco.

⚠ Recomendamos que o certificado de produção seja utilizado exclusivamente neste ambiente e que **NÃO** seja utilizado no ambiente de homologação

TEMPLATES

1. Cadastro de *Client ID*

Para cadastro de credenciais e certificado de novos parceiros em ambiente de *sandbox* e produção, use o *template* abaixo:

- No campo assunto do e-mail deverá ser preenchido:
    - CAD-HML | NOME DA API | RAZAO SOCIAL | CNPJ
    - **Exemplo**: CAD-HML| PIX | EMPRESA PARCEIRA LTDA | 00.000.000/0000.00

⚠ No caso de produção, seguir o mesmo exemplo acima, alterando apenas o texto de CAD-HML para **CAD-PRD**

⚠ Se o CNPJ de homologação for **diferente** do de produção, o CNPJ do ambiente deverá constar no assunto

⚠ Atentar-se ao padrão brasileiro de CNPJ, com **14 dígitos**

- Anexar o certificado (**chave pública**) seguindo as regras abaixo:
    - **Formato** .cer, .crt ou .pem
    - **Validade** mínima de 4 meses e máxima de 3 anos
    - Para ambiente de homologação serão aceitos **certificados auto assinados**
- Enviar no conteúdo do e-mail as seguintes informações:
    - **Breve descrição/uso** da aplicação consumidora
    - **Produto (API) ou serviço** que será consumido
    - **Número de celular** (de preferência com WhatsApp)
    - Dois e-mails de **contatos de referência** para renovações de certificados e avisos

2. Solicitação de Suporte Técnico e dúvidas

Em caso de dúvidas para **conectividade** com nossas APIs, entre em contato pelo e-mail: suporte.api@bradesco.com.br.

Para acionamento do suporte favor enviar o e-mail nos padrões abaixo:

2.1 Para dúvidas técnicas de conectividade de API em homologação

- O **assunto** do e-mail deve seguir o padrão:
    - DUV-HML| NOME DA API | RAZAO SOCIAL | CNPJ
    - **Exemplo**: DUV-HML| PIX | EMPRESA PARCEIRA LTDA | 00.000.000/0000-00
- Contextualizar no corpo do e-mail a dúvida com o máximo de **informações** possíveis:
    - **URL** utilizada pelo cliente para acessar o serviço
    - **Data e hora** da requisição
    - **Body** da requisição (ou cURL da chamada)
    - **Headers** da requisição
    - **Body** do response

2.2 Para dúvidas técnicas de conectividade de API em produção

- O **assunto** do e-mail deve seguir o padrão
    - DUV-PRD| NOME DA API | RAZAO SOCIAL | CNPJ
    - **Exemplo**: DUV-PRD| PIX | EMPRESA PARCEIRA LTDA | 00.000.000/0000-00
- Contextualizar no corpo do e-mail a dúvida com o máximo de **informações** possíveis:
    - **URL** utilizada pelo cliente para acessar o serviço
    - **Data e hora** da requisição

- **Body** da requisição (ou cURL da chamada)
- **Headers** da requisição
- **Body** do response

1. Criando um certificado auto assinado

Para criar certificados auto assinados, a ferramenta que costuma ser a mais usada se chama OpenSSL. Em ambientes Linux ela costuma vir instalada como padrão. Ao executar os comandos são gerados dois arquivos, um sendo a chave pública (cert.pem) e o outro a chave privada (key.pem).

⚠ A chave privada **nunca** deve ser compartilhada, pois é ela que garante a autenticidade da chave pública, que será compartilhada conosco.

É necessário que o campo CN do certificado contenha a razão social e CNPJ da empresa, pois essas informações serão usadas para fins de validação interna.

```
openssl req -new -x509 -sha256 -newkey rsa:2048 -nodes -keyout parceiro.homologacao.key.pem -days 365 -
out parceiro.homologacao.cert.pem -subj "/CN=RAZAO SOCIAL:CNPJ/C=BR/ST=SP/O=RAZAO SOCIAL"
```

Na imagem acima temos um exemplo de como o comando deve ser executado. Onde o termo “parceiro” aparece, você pode substituir para um nome de sua preferência para fins de organização.

Quando o comando for executado, serão gerados dois arquivos:

- Chave privada (parceiro.homologacao.key.pem)
- Chave pública (parceiro.homologacao.cert.pem)

A chave pública deverá ser compartilhada com o Bradesco e a chave privada deverá ficar sob sua tutela e não ser compartilhada com ninguém. As validações de autoridade serão feitas através da comparação dessas chaves.

2. Gerando token de acesso em homologação (*access token*)

Para acessar uma API do Bradesco no ambiente de homologação, você deve gerar os tokens a partir das URLs abaixo:

- https://proxy.api.prebanco.com.br/auth/server/v1.1/token
- https://proxy.api.prebanco.com.br/auth/server/v1.2/token

As requisições deverão conter um **JWS** no ***body***. Um JWS, de forma direta, é um JWT assinado com uma chave privada. No seu caso a chave privada a ser utilizada para assinar o JWT será àquela criada no momento de cadastro das suas informações para geração do Client ID. Como você nos enviou a chave pública que era par da chave privada, conseguiremos validar a sua autoridade.

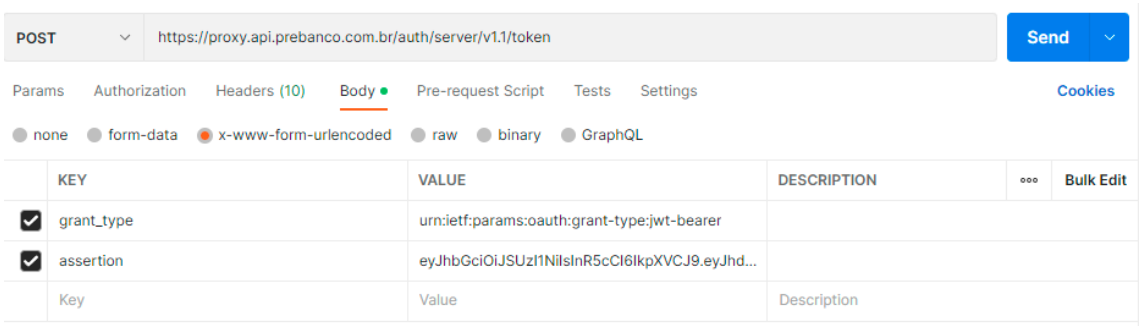
Se você precisar de ajuda para criar um JWS, veja o guia de referência "**Criando um JWT assinado (JWS).**

⚠ A URL **correta** para o seu caso dentre as duas será informada no e-mail contendo o Client ID/Client Key previamente criado.

⚠ Para o ambiente de **produção**, a URL será sempre "/auth/server/v1.1/token".

A requisição deverá conter os seguintes itens:

- **Método**: POST
- **URL**: https://<endereco_do_ambiente>/auth/server/v1.1/token
- **Headers**
  - **Content-Type**: application/x-www-form-urlencoded
- **Body**
  - **grant_type** : urn:ietf:params:oauth:grant-type:jwt-bearer
  - **assertion**: <JWS gerado>



Ao rodar a chamada assim, será obtido o seguinte resultado:


```
{
    "access_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzUxMiJ9.ew0KICJ2ZXIiOiAiMS4wIiwNCiAiaXNzIjogImh0dHBzOi8vcHJveHkuYXBpLnByZWJ
hbmNvLmNvbS5ici9hdXRoL3NlcnZlci92ZMS4xL3Rva2VuIiwNCiAiYXVkIjogImh0dHBzOi8vcHJveHkuYXBpLnByZWJhbmNvLmNvbS5i
cjo4NDQzIiwNCiAiaWF0IjogMTY3NjQ3MTM4MCwNCiAiZXhwIjogMTY3ODg4ODgyOCwNCiAic2NwIjogIkNSSU0sUFNEQyxQU0NFLFVPU
kcsUkVDQSxwYWd0b3MscGdib2xldG8sR0FEQyxBdGVuZGltZW50byxURURCT1BFTixTUEksUkVTVCxGQUNFTUFUQ0gsQVRNLExHUEQsRk
lOSSxBVE0sR0NDTixQQ0NGLEJDUEEsQ0JPTixFUE1GLERCVE8sQUdSTyIsDQogImp0aSI6ICJtb2NrU2Vzc2lvbjMwMTY0NzYxMTg4NTI
yMzU0ODkzOTQyMTM5MTg2NzgyMDI0NjU1IiwNCiAidG9rZW5UeXBlIjogImFjY2VzcyIsDQogImNsaWVudFR5cGUiOiAic2VydmVyIiwN
CiAib3duZXJUeXBlIiA6ICJzZXJ2ZXIiLA0KICJhdXRoRGF0YSI6ICJleUowZVhBaU9pSktWMVFpTENKamRIa2lPaUpLVjFRaUxDSmhiR
2NpT2lKU1UwRXRUMEZGVUNJc0ltVnVZeUk2SWtFeU5UWkRRa010U0ZNMU1USWlmUS5hQTRhdzlpRFpBRVZ3cXJtZnhpSm53Sndyc0xBeF
NKbFBXemUxWkZfOUhyWVgzNG9UeFY0SHlJRjhsQW1pM1U1SE5zSkItaHJZNU5LRVk2RzV2RlFhV3ZneWR1TGprb295SDRyQWZVa2RmRXB
PUE1QY2otUERMVm16MGVQbC1xOXE2T3ZIZHlZEJfbGdfdkkySjdUanVieDRMWktDTWItdlphdUtuMFp3STNkY2c0cEZ6bEZUSnBsRVE1
Z3k3OWtvUTI1S3NjQlhMbDVFNVc1RlZtQnVkUXFmZHg4ZmVfVjFpOV8yMDNIbEVMRllvUGdkOURTVUo3b0lKZjBtZ2xpLVJzVl9Ha1Qwa
FVpdjdTWXEzOXN0RnV6TWhCTEktVjdPa3BxMUJQc05neFp0YmJQT2k2ZVBhNG5Qb0FIcmVNOXd6NGg2X0lxMi01UEJBdkVxakZwRGZEQm
01Y1hVZmpzNGpldnZGeC1RbHhFVTRJU3pqRmlrTUVtNXRxLVVnTWpXYUN2cG1mU1Nwd0t3SUxRQUtMNWpxVVFpVmxvTWFhY1N6MzkyX0p
SckZzMzdFakFGMjRkVnAxWlNOMXpyODd4LW41YUdKQ0U2SExUQ0h6RS03Z0k1UkF2UEFVZmlPWWpiSzgyUWdnNlY5UjFvNEtnNlJYcUFH
bnBaamd6NFlQZXZad0owbGNHc1VMR0ZJQkh5b0IwSUdqN25RR2xiVWJhSWw2UE1DTGNhOHNyTks1VERhQW03NXUtR2xPRVpESHJSV2pza
01PTlN1RHlfX0EtVGxKa3hFdDdOYUNyRlhmdHZlSUtzM0ZGVTV0SzJTT2EwaHM5eW5aYnJmR3FPN3U4QjdhMzlBcFZZX1Y5Ym41RzFWNT
V6SzZraG9IVDVvdmp3N1dod2ZPbWpFTS5CbHVqNFN2VlN5WGQ0Y2kycmRwSFF3LnAweVRrUlU0Wk1BRU1hQjhjbDZJMGtyVkpkbmRaMFY
5R2RTeGpKcFpPeWFNUGgxcTlqWTVxaGlfZHYtVlQ5TVRnUTFQQ3lEMWVLZ1RHd1BDNTJtUXBTRmU0UnJGbDlTaTg4c19DYlEwSTZKUF93
UlFKc2daQWRZRmJ3MDF5RGs3TDFEUEZYNElvWF9ZbDFjeF9sMWRlRGh4RGNVWnRvQXotUGItQjZscDBjLVdzYjI0Qi1qRjVuRkZ3NU1GR
GUydjhNQ085R1JoY2VkMXVEaE1JTWVESC01ZHJEeTFWQ0Rad3BiQ2FIZDJtMDl2S18zQXljQllkUHRSbGVUdFVKakZncjdEUU9vSV95UD
B6b2R3NmU4THZBYWdSRkxGTmVoTmFJY3ZGUXVLYlZYWGVhUF9EYWw4S1dXWHNZQlNrLXplMnB6SUxXWURXaFNkcU9yVU5JOERTV25kU2R
QdmUyQWhCeTlsY0RzQWt0eXNpY0t5YUhvWUsyelBKY0tPUEVINzdCTkdGVkdmeVhCQWRkeGNKaXIyZWlidUQycmVtd3oyZE1NNE1qOGJ5
LVJjTGw1MDJncEJmbm1ReGlMMXVNaTJBQ1FmNW5leUtOTzViSjN3dF9vYjlrbWoyUkgtWHVDMklDU1JJem1XaGZtekNwcUU2ci0zRzI0U
Dl0akw1RlMtWVI4eDRFaDVlTnFFbjdhZEhTUGQ5TXpGTWtCa01MSEJOY0RQX2hhTWZMb1pkbmZqRnVyR1V5eWNnTS1JMXp5XzdCdzFubG
9DTk83WUc5d002eTNwTlp0RlN4a3hybXFaNFp1X3FRUFA3TkFzZncxaW5UXzI2SWVoNEZOVVFnZEtlT29fclNJd3dfTmdjLTdya3dnSk9
5TlVxV2VOM0lheDltZzBKa00zSzMzeWU3N0paQl95aXhJN3p6Uy11MTU5bnFXMU4wazBDU0w2eHdyOTl6OVozZUE3R0syN0xuSF9Od2h1
VUJHdTZ3RHFfMzl6emlfMnJ5eTJMUXh0akNMUmtlWl9mN0lFWjdKSFhCWVI3eGJZT2NVanF0MVRNdmI5YmR6NUttQ3VVWWtSRWVWNXFfU
EUwbjNtU1A5WkZVSmxCOWJyVFhJcTE0RlBXbERXVUgyTXJwOWo1NW1NdWRFdmtwOW9sdVFHeXItSkE5ZkkzTlNxZmcyNnkzOGZQZHJfb3
M2QnM0VVdfdUZpcFNidXJtMU5aZVk4Yl9iNmpSeUNESFFaT2hmVXZ6RUlUd25SU1loTE9HdnFTTkxUVFh4NlJkNmdqQU1FWk5SM2laNnJ
ydjNoVTNvdURKbURUYYXcwaGhwVElVZjNuNWFpTW9Yai14ZVBIMFhnUWRTVHJqQm9xMDg4d2E4dlU2QmFpV3lNRDl1YldUaFlmNDZfMmZN
NXQteFR0X2NOUGhSLXJqcDZoXzNRSEV3UDFPVXhzM1NoY0g1TFRBbHFqeWRqTElzWVppcE1FNVRJTEZjMS1pakkwUTZ0ZzBjRGJraGlyN
lJlRUY0M0ZKbGJEcHNySTkyY0hCd08xUVZRclRsblUzODNrcGlzaHBSemU0T3FpcGdFMFA0MGFnOXJHU0hlbGFDT3NtcnNtanV6TmRmM2
p2UEFfcng0RXJXb0I0S1NZSGNRYnZmZTdzNS1RQ2hxLWwxd2M1LUphRTc4bGRXQVNYd0ZwcEhxWENVOU1YTXRkQVk1ZFVDWGNFRlZHNnF
vWUk2eTc4cTFtTy13MTRpZUhsNTZ3ZDJBZS1vSUkwNjlBbGNMcFpzM0dTd29fVGRqVHJpQzJkUzVfWVZBbnYtd3hKUGk2Vjd4QXFEQ1Vy
enAzTE9NcmJhai11dnVCOHpPSzVWOEZXY3J6LVhpNXdXNFFvenBNYnpBRWNLcFFHZXVNckw4dkc4dmNPWkJqZ2NlT0xUbTcxXzdMajlFL
Ws0LlRQdTlwbFBmN3dSdU1mNmZMY2pYaDRVNDJaQUhiWDd0RkYtamxvR1N0b3MiDQp9.rm2jesPm-
1_C1gz5LHVYZEdagD0Zt39CArdEi6qD4BzIDDBAhM5ctbuAMH1zfiy-S8QT1Kc52Hw4tXo6d-HACKRU7df8VLnqmJm-
t7o9_TyipFKu6LHcpA-
Lh2DROIhOIEuHZMZux2trlfRGrfjGS8qDZ79Wog64jgim6fr_L5mLwEQZEZs7ngH71F2kB8zhJOpXCCrDxg_5tiKx5qggf344vk5ew9DW
GU6gp5ovAKvfrIaaxyhkrZ_9MMzneHUa2N-xNMCr-omhdMqf3-
opVMHiJ839ZRyRt7nx6tU6vQV3TfCmHD7GAgX0DwkU5jT8Sr2uHRTv2PxdjKoXQmmJiF8lm0DRWBMDmburDEhn96CRujW2VnBaL2A8fkI
TSGjx9V5K9rVhOwTUtjgrueibK0-
__bCfm3KfZrR84odzD3j0UFWsVR5i4E0aSmYKVXYZmPKJn3ZvzMRSwns4YbbQqIkGjpBYhd2KNRTeK0hIxZh8csoc2OgF97hzGiJMhrXY
mz43gAGJIF6_oGddVpmJoKJ9mCbUSa6j6yeXgWIG24h_JfNOxxrYQH5kd9-QxJhgjcMRbMFhs90RBozLABirO-8692-4n-
Qqi42V0OEEKlJH9y4p6GIZdaee8wm85mIn87Goqri9P-qfd2pG6qizVkNm4knMHlSFuqU2NH0kjEs",
    "token_type": "Bearer",
    "expires_in": 2417446
}
```

O tempo de expiração de *access token* deve ser controlado pela aplicação e terá validade de uma hora. É recomendado que o mesmo *access token* gerado seja reutilizado em todas as requisições aos *endpoints* de serviço durante este período.

3. Criando um JWT assinado (JWS)

Antes de falar sobre o JWS, precisamos definir o que é um JWT. O JWT é uma estrutura JSON formada por três partes: um header, um payload e uma assinatura. Cada um dos três itens são uma estrutura JSON apartada. Existe um padrão para cada um desses campos, sendo descrito a seguir:

a. Header

Objeto JSON contendo os parâmetros que descrevem as operações criptográficas e os parâmetros empregados. O cabeçalho JOSE (JSON Object Signing and Encryption) é composto por um conjunto de parâmetros de cabeçalho que normalmente consistem em um par nome/valor: o algoritmo de hash sendo usado (por exemplo, HMAC SHA256 ou RSA) e o tipo do JWT.

b. Payload

O *payload* contém campos que também são conhecidos como *claims*. Esses campos incluem declarações de segurança verificáveis, como a identidade do usuário e as permissões permitidas.

- O campo ***aud*** sempre possuirá a URL de *token* completa que está sendo utilizada para o ambiente.
- O campo ***exp*** deve ser definido adicionando-se uma hora ou um mês ao atual valor do ***iat***
- Se o ambiente utilizado for **homologação** o "***aud***" deve ser informado:
    - "***aud***": "https://proxy.api.prebanco.com.br/auth/server/v1.1/token
- Se o ambiente utilizado for **produção** este campo deve ser:
    - ***aud***": https://openapi.bradesco.com.br/auth/server/v1.1/token
- No caso de alguma API em homologação necessitar que o Token seja gerado na URL https://proxy.api.prebanco.com.br/auth/server/v1.2/token, o "***aud***" do *payload* deve ser mantido com a informação padrão https://proxy.api.prebanco.com.br/auth/server/v1.1/token

c. Assinatura

Para assinar um JWT e transformá-lo em um JWS, primeiro precisamos transformar o *header* e o *payload* em Base64 - isso pode ser feito até em alguns sites na internet - e concatená-los comum ponto (.) no meio

Um exemplo:

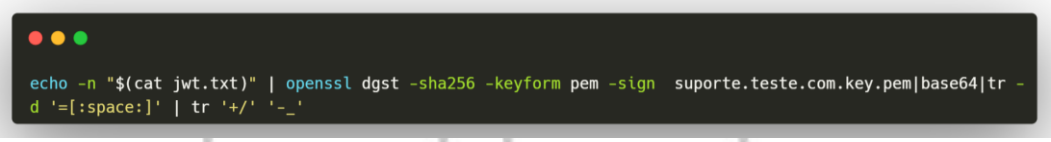
O *header* e payload abaixo formam o seguinte JWT:

Header:

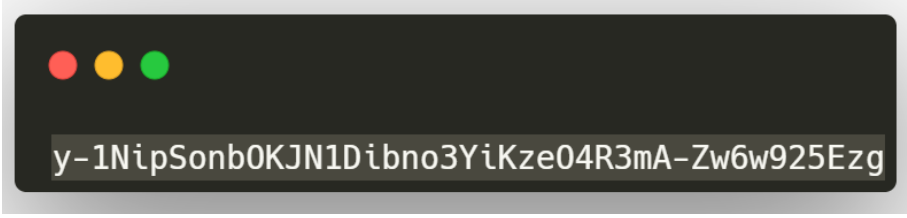
Payload:

JWT:

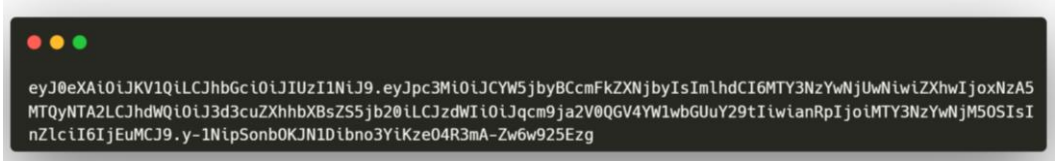
Para a assinatura ser montada é necessário usar a chave privada. Para isso podemos usar a ferramenta OpenSSL. Utilize o comando abaixo para obter o JWS.



Esse comando irá gerar a string de assinatura, conforme abaixo:



Agora basta colar essa parte final no seu JWT para que ele seja um JWS.



Após criar o JWS, será possível solicitar um *token* JWT para acessar as APIs, conhecido também como JWS.

4. Criando o header *X-Brad-Signature* para consumo de APIs

O header ***X-Brad-Signature*** é um campo obrigatório para todas as chamadas de consumo de nossas APIs. O seu valor é um JWS que deve ser assinado com os seguintes campos:

- Método HTTP usado na requisição
- *Endpoint* da chamada
- *Query Parameters* caso existam
- *Body* da requisição ou deixar uma linha em branco nos casos em que não existe *body*
- *Token* de acesso gerado anteriormente
- *Nonce* (valor numérico de no máximo dezoito dígitos, só pode ser usado uma vez)
- *Timestamp* representando o momento da chamada (o mesmo usado no header ***X-BradTimestamp***
- Algoritmo usado para assinar o JWT, no campo *header* ***"X-Brad-Algorithm"***

Todas essas informações devem ser escritas em um arquivo texto simples, uma linha abaixo da outra, com o arquivo sendo assinado ao final do preenchimento desses dados. A *string* gerada nessa assinatura deverá ser usada no campo ***X-Brad-Signature***.

```
POST
/v1.1/jwt-service
agencia=552&conta=331
{"teste":"valor"}
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzUxMiJ9.ew0KICJ2ZXIiOiAiMS4wIiwNCiAiaXNzIjogImh0dHBzOi8vcHJveH
1672145026618
2023-02-15T11:38:00-00:00
SHA256
```

Exemplo de como o arquivo de texto deve ser preenchido

```
echo -n "$(cat request.txt)" | openssl dgst -sha256 -keyform pem -sign  suporte.teste.com.key.pem  |
base64 | tr -d '=[:space:]' | tr '+/' '-_'
```

Comandos usados para gerar JWS que será enviado nas chamadas de API

GLOSSÁRIO

Access Token
Token de autorização usado para acessar recursos protegidos em um serviço ou aplicativo.

Autenticação
Processo de verificação de identidade de um usuário ou sistema antes de conceder acesso a recursos protegidos.

Autorização
Processo de permitir ou negar o acesso de um usuário ou sistema a recursos específicos após a autenticação.

Base64
Método de codificação para representar dados binários em ASCII para transmissão em sistemas que não aceitam dados binários diretamente.

Body
Parte da mensagem HTTP que contém os dados transmitidos na requisição ou resposta.

Certificado digital
Arquivo eletrônico que contém informações de identificação e autenticação para um usuário, dispositivo ou aplicativo.

Chave pública e privada
Par de chaves criptográficas usadas para autenticar usuários ou dispositivos e criptografar dados.

Client ID
Identificador exclusivo atribuído a um aplicativo cliente registrado em um sistema de autenticação e autorização.

Client Key
Chave criptográfica privada usada para autenticar um aplicativo cliente em um sistema de autenticação e autorização.

Endpoint
URL que um cliente pode acessar para interagir com um serviço ou aplicativo.

Header
Parte da mensagem HTTP que contém informações sobre a requisição ou resposta.

JWT
JSON Web Token, um padrão aberto para transmitir informações seguras como tokens de autenticação ou autorização em um formato JSON.

JWS
JSON Web Signature, um formato para assinar digitalmente informações JSON.

OAuth 2.0
Protocolo de autorização para aplicativos de terceiros que permite o acesso seguro a recursos protegidos de um serviço ou aplicativo.

Open API
Interfaces de programação de aplicativos abertas e disponíveis publicamente para acesso e integração.

Protocolo mTLS

Autenticação de Transporte de Camada Dupla (mutual TLS), onde tanto o cliente quanto o servidor autenticam uns aos outros com base em seus certificados digitais.

Payload
Dados transmitidos na parte do corpo de uma mensagem.

Query Parameters
Informações adicionais transmitidas na parte da URL de uma requisição HTTP que podem ser usadas para filtrar ou modificar os resultados retornados.

Requisição
Mensagem enviada por um cliente para solicitar uma ação a ser executada por um servidor.

Resposta
Mensagem retornada pelo servidor em resposta à uma requisição feita por um cliente.

Token
String de caracteres que representa a autorização concedida a um usuário ou aplicativo para acessar recursos protegidos.

URL
Endereço da Web que identifica um recurso específico que pode ser acessado por um cliente.