

Manual de Integração Averbeporto Porto Seguro

Manual de Envio de XML para Averbação

O sistema de Averbação Eletrônica da Porto Seguro, o [Averbeporto](#), recebe os arquivos XML de [MDF-e](#) (58), [CT-e](#) (57), Minuta de CT-e (94), [NF-e](#) (55), [NFC-e](#) (65), [DI](#) e seus XMLs de evento de cancelamento, para automatizar o processo de [Averbação de Seguros](#).

Os arquivos XML podem ser enviados manualmente via Interface Web ou automaticamente, via E-mail ou API (Webservice REST). Em todas as modalidades, os arquivos XML podem ser enviados 1 à 1 ou em lote, compactados em um arquivo ZIP.

Nas telas de **Busca de Arquivos/Exportar para relatório** ou **Relatório/Registro de Envios** é possível consultar o **Número de Averbação ANTT** para os documentos guardados. No tópico [Consulta Número de Averbação ANTT](#), estão descritas as formas sistêmicas de consulta.

Nota - 04/2022

Aos usuários que já utilizam a API, observar que o endpoint correto deve iniciar com **apis** ou **api** - Por motivos técnicos, o acesso via **www** ou **wws** será desativado em breve.

Não utilize a senha de acesso web para acessar a API, pois o acesso à API com ela será desabilitado no futuro. Gere uma senha de API própria no módulo **Cadastro do Usuário**.

Para aqueles que fazem averbação via e-mail para CNPJ@averbeporto.com.br, não enviar como Cópia Oculta Cco (Bcc).

Sumário

Controle de Versões	2
Manual de Integração atualizado	2
Averbação via Interface Web	3
XML	3
ZIP	3
Averbação via API (Webservice tipo REST)	4
Averbação via E-mail	7
Consulta Número de Averbação ANTT	8
Acessando o API (Webservice REST)	8
Acessando a Página/URL de consulta	8
Parâmetros Adicionais de Averbação	10
Acesso via Postman	13
Exemplo em PHP (cURL)	14
Exemplo em Delphi (Indy)	18

Controle de Versões

Versão	Produção	Mandatário	Descrição
1.0	29/01/2015	--	- Versão inicial
2.0	29/08/2017	01/10/2017	- Versionamento de API (v=2) - Número de Averbação ANTT (MDF-e 3.0)

Manual de Integração atualizado

Google Docs: <https://goo.gl/FWAvBE> ou <https://averbeporto.page.link/manual>



Averbação via Interface Web

1. Com seu navegador web, acessar o endereço <https://wws.averbeporto.com.br>
2. No canto superior direito, **fazer login** com seu usuário e senha
3. No menu à esquerda, clicar em **Envio de Arquivo**
4. Na tela, selecione seu arquivo XML ou ZIP e clique em **Enviar e Guardar**
5. Nas telas de **Busca de Arquivos/Exportar para relatório** ou **Relatório/Registro de Envios** é possível consultar o **Número de Averbação ANTT** para os documentos guardados. [[Sobre outras formas de consulta](#)]

Para mais informações sobre o sistema web, consulte a **Ajuda** no menu à esquerda.

Os arquivos XML e ZIP devem seguir o padrão de codificação para [tipos de mídias oficiais](#):

XML

- [application/xml](#)
- [text/xml](#)

ZIP

- [application/zip](#)
- Limite de ~4MB de tamanho (< 1.000 XML).

Averbação via API (Webservice tipo REST)

Resumo: Acessar a [API](#) (RPC) via [método POST](#) para **Login** (mesmo Usuário do [Sistema Web](#) - que deve ser

alterada caso tenha menos de 8 dígitos; Senha da API deve ser gerada no módulo **Cadastro do Usuário**) e receber o **json** de resposta (também contendo “ses”) e o **cookie de sessão (portal[ses]** - HEADER HTTP - deve ser utilizado em múltiplos envios), acessar novamente para **Enviar o arquivo** e receber o **json** de resposta que irá conter o status e o Número de Averbação ANTT. O **Sistema Web** é um programa em **JavaScript** acessando esta mesma API. Ou seja, todas as requisições e respostas podem ser observadas via **Ferramenta do Desenvolvedor** em seu Navegador Web.

1. Endereços da API *¹:

Seguro: <https://apis.averbeporto.com.br/php/conn.php> (:443 - **TLS** 1.0, 1.1, 1.2 e 1.3) *² - **HTTP/3**

Plain: <http://api.averbeporto.com.br/php/conn.php> (:80 - Plain text - **HTTP/2**) - **descontinuado**

*¹ **NÃO utilizar www. ou wws. ou nenhum outro subdomínio para acesso à API, será bloqueado.**

*² Pode ser necessário definir um user-agent como “Mozilla/5.0 (Windows NT 6.1; WOW64; rv:12.0) Gecko/20100101 Firefox/12.0” para acessar [apis](#), isso impedirá que o CF bloqueie seu programa com o [erro 403/1010](#)

*³ Chamar o endereço da API no navegador irá gerar uma mensagem de erro (item 7) pela absoluta falta de parâmetros na comunicação.

2. Para **login**, POST (application/x-www-form-urlencoded) os parâmetros:

```
{
  "mod": "login",
  "comp": 5,
  "user": "USUARIO_CNPJ",
  "pass": "SENHA_Credencial_do_Web_Service_API",
  ["dump": [1,2]] // Opcional.
}
```

- **dump:** Pode ser utilizado em qualquer requisição à API para auxiliar na depuração. Exibe um “dump” de como estão chegando as variáveis enviadas ao sistema pelo usuário.

dump=1: Adiciona a tag “dump” no json de resposta, após todo o processamento normal da requisição.

dump=2: Exibe o json de resposta apenas com a tag “dump”, antes de qualquer processamento, e aborta o processamento normal.

- **pass:** **Não utilize a senha comum para acesso web no acesso à API. Acessando o [sistema Web](#), vá no módulo **Cadastro do Usuário** e gere uma senha de API própria, clicando no **X** para iniciar o processo.**

3. Json de falha no login, usuário ou senha inválidos, enviados de maneira incorreta, ou sessão de login expirada:

```
{
  "success": 1,
  "logout": 1
}
```

Json de login bem sucedido:

```
{
  "success": 1,
  "C": {
    "id": "00",
    "userName": "USUARIO",
    "name": "Usuario",
    "email": "usuario@dominio.com",
    "portal_groups_id": "00",
    "type": "U"
  },
  "S":[...]
}
```

4. Para **upload**, fazer o POST (multipart/form-data) do **cookie**¹ de sessão (**portal[ses]** - HEADER HTTP - recebido no login), do **arquivo (file)**² e dos parâmetros:

```
{
  "comp": 5,
```

```

"mod": "Upload",
"path": "eguarda/php/",
"recipient": "",
["v": N] //N = Versão da API. Parâmetro opcional.
}

```

Obs 1. Os COOKIES residem no cabeçalho da resposta HTTP e não no corpo como o JSON.

Obs 2. Enviar o arquivo junto aos parâmetros, mesmo como um parâmetro codificado em base64, não irá funcionar. O arquivo deve ser enviado da mesma maneira que um formulário HTML o faria, em multipart.

Obs 3. Quando gerada a Credencial de API no módulo Cadastro do Usuário, o cookie de sessão passa a ter validade de 1 ano, ao contrário da 1 semana padrão.

5. Opções do parâmetro recipient (opcional, utilizar somente se assim indicado):

Parâmetro que indica o tipo do(s) arquivo(s) sendo enviado(s). Necessário apenas nos casos indicados pela Seguradora.

Vazio = Automático (Recomendado na grande maioria dos casos)

E = Embarcador Emitente (Em desuso)

F = Fornecedor (Em desuso)

T = Transportador (Em desuso)

D = Duplo Ramo (As NF-e são averbadas 2 vezes, uma como T e outra como E)

6. Json de retorno:

```

{
  "success": 1,
  "S": [
    "P": 1, // Processado (xml guardado com sucesso)
    "D": 0, // Duplicado (xml pré-existente)
    "R": 0, // Rejeitado (xml não parece ser do tipo certo)
    "N": 0 // Negado (Não é xml ou zip)
  ]
  "prot": "1234567890123", // Protocolos do XML guardados P
  "prot": ["1234567890123"], // Protocolos dos XMLs (ZIP) guardados P
  "error": [ // Mensagens de erro para os resultados R ou N
    "code": 00,
    "msg": "Mensagem de erro"
  ]
}

```

7. Json de erro genérico, significa “Erro desconhecido” e, geralmente, é resultado de uma requisição mal formada ou efetuada via um método diferente de POST:

```

{
  "success": 0,
  "error": {
    "msg": "Ghs.loc.error"
  }
}

```

8. Note que, no json de resposta, um “**success**”: 1 se refere ao sucesso na comunicação, não necessariamente a efetivação da requisição.

9. Em caso de falha no arquivo ZIP, deve-se consultar o **Relatório de Envios** no sistema web para verificar os arquivos que foram guardados.

10. Nas telas de **Busca de Arquivos/Exportar para relatório** ou **Relatório/Registro de Envios** é possível consultar o **Número de Averbação ANTT (Protocolo de Envio)** para os documentos guardados. [[Sobre outras formas de consulta](#)].

- O envio tem as mesmas características de um POST de formulário HTML ([mídia application/x-www-form-urlencoded](#) ou [multipart/form-data](#)).

● Exemplos CURL

Dicas: -i mostra o response header; -v mostra toda a comunicação; --tls-max 1.2 limita a versão do tls

Login:

```
curl 'https://apis.averbeporto.com.br/php/conn.php' --compressed -c cookie.txt -X POST -H 'Content-Type: application/x-www-form-urlencoded; charset=UTF-8' --data 'mod=login&comp=5&user=USUARIO&pass=SENHA'
```

Upload:

```
curl 'https://apis.averbeporto.com.br/php/conn.php' --compressed -X POST -H 'Cookie: portal[ses]=8c878b3ab2ba27bf7bc4d5d448b6489b' -H 'Content-Type: multipart/form-data' -F mod=Upload -F comp=5 -F path=guarda/php/ -F file=@ARQUIVO.xml
```

ps:

USUARIO e SENHA devem ser substituídos pelo usuário (CNPJ) e senha do Segurado (usuário e senha próprio para acesso à API do [Sistema Web](#)).

O Cookie deve ser substituído pelo cookie retornado no login.

O ARQUIVO.xml deve ser substituído por um documento XML (corretamente formatado) do Segurado.

Averbação via E-mail

O envio via [E-mail](#) tem limitações técnicas inerentes, tanto na velocidade que é possível enviar XMLs quanto na leitura deles pelo sistema Averbepo. Se o seu sistema for enviar centenas de XMLs por dia, ou tem necessidade de receber o protocolo ANTT imediatamente após o envio, esse método não é recomendado e, nesse caso, **o melhor é realizar a integração via [API](#)**.

1. No caso de aprovação da Porto Seguro em aceitar as averbações via e-mail, uma conta de e-mail no formato CNPJ@averbeporto.com.br será disponibilizada.
2. Essa conta deve ser usada como destino apenas de mensagens com anexos de documentos eletrônicos no formato XML.
3. Para que o sistema seja capaz de averbar esse arquivo XML, ele deve estar anexo diretamente na mensagem ou anexo em formato ZIP (XML dentro do ZIP - ZIP dentro de ZIP será rejeitado, assim como qualquer outro formato diferente destes).
4. Limite de ~4MB de tamanho (< 1.000 XML).
5. Nas telas de **Busca de Arquivos/Exportar para relatório** ou **Relatório/Registro de Envios** é possível consultar o **Número de Averbação ANTT (Protocolo de Envio)** para os documentos guardados. [[Sobre outras formas de consulta](#)].
6. **IMPORTANTE:** Ao enviar mensagem para para sua conta de e-mail CNPJ@averbeporto.com.br, não enviar como **Cópia Oculta Cco (Bcc)**. O destinatário deve, obrigatoriamente, estar em **Para (To)** ou **Com Cópia (Cc)**.

Obs. Caso já esteja utilizando a averbação via e-mail e integre seu sistema via API posteriormente, solicitamos que os envios via e-mail sejam imediatamente interrompidos, pois o envio dos XML em duplicidade pode causar inconvenientes em suas averbações.

Consulta Número de Averbação ANTT

Resumo: Acessar a API via método POST ([assim como no módulo Upload](#)) ou acessar a URL via método GET e receber a resposta da consulta.

O “[Número de Averbação ANTT](#)” (Protocolo de Envio), advindo de um CT-e ou NF-e, deverá

necessariamente ser utilizado para a composição de um MDF-e que os carregue (a partir da versão [3.00 do MDF-e](#), de acordo com resolução [ANTT 4799/2015](#)).

Acessando o API (Webservice REST)

1. Endereço da API: <https://apis.averbeporto.com.br/php/conn.php>
2. Para **consulta**, fazer o post do **cookie** de sessão (**portal[ses]** recebido no login), e dos parâmetros:
comp=5&mod=Protocolo&path=atwe/php/&chave[]=12345678901234567890123456789012345678901234&chave[]=2234567890123456789012345678901234

Neste exemplo estão sendo pesquisadas 2 chaves. Para realizar a pesquisa inversa, basta passar os protocolos nos parâmetros **protocolo[]=1234567890123456789012345678901234567890** e omitir os parâmetros **chave[]**.

3. Parâmetros opcionais:

out = "json", "xml" ou "csv" // Formato do retorno. Padrão é Json (Na forma padrão da API)
download = 0 ou 1 // Cabeçalho do retorno como display ou download para arquivo. Padrão é 0 (display)

delim = , // Parâmetro para out=csv. Indica o delimitador de campo do CSV. Padrão é vírgula (,)

4. Json de retorno:

```
{
  "success": 1,
  "S": [{
    "chave": "12345678901234567890123456789012345678901234",
    "protocolo": "1234567890123456789012345678901234567890"
  }, {
    "chave": "22345678901234567890123456789012345678901234",
    "protocolo": "2234567890123456789012345678901234567890"
  }
]
```

Acessando a Página/URL de consulta

1. Página de consulta (**standalone**) - Gera URL GET:
<https://wvs.averbeporto.com.br/atwe/protocolo.html>
 2. Exemplo de acesso via GET: [https://apis.averbeporto.com.br/atwe/php/Protocolo.php?out=json&download=0&chave\[\]=12345678901234567890123456789012345678901234](https://apis.averbeporto.com.br/atwe/php/Protocolo.php?out=json&download=0&chave[]=12345678901234567890123456789012345678901234)
- Parâmetros opcionais:

out = "json", "xml" ou "csv" // Formato do retorno. Padrão é Json (Na forma padrão da API)
download = 0 ou 1 // Cabeçalho do retorno como display ou download para arquivo. Padrão é 0 (display)

delim = , // Parâmetro para out=csv. Indica o delimitador de campo do CSV. Padrão é vírgula (,)

3. Exemplo de retorno CSV:

Chave,Protocolo

12345678901234567890123456789012345678901234,1234567890123456789012345678901234567890123456789012345678901234567890

4. Exemplo de retorno XML:

```
<data>
<item0>
<chave>12345678901234567890123456789012345678901234</chave>
<protocolo>1234567890123456789012345678901234567890</protocolo>
</item0>
</data>
```

Parâmetros Adicionais de Averbação

1. Campo Texto de Observações

O Sistema pode também ler **informações adicionais de averbação** passadas de maneira especial na tag de **observações (xObs ou infCpl)** dos documentos eletrônicos.

Para **inserir informações adicionais** de averbação no campo de **Observações** de seus documentos eletrônicos, use o **Gerador de Parâmetros de Averbação** em <https://www.averbeporto.com.br/atwe/obs.html>.

Após gerar o código de informações, **basta copiá-lo e colá-lo** em seu sistema TMS, no campo de observações do documento eletrônico.

Exemplo de utilização das Observações:

```
{mudanca:"s",vcontainer:"1.23",viagem:"1",placa:"ABC1234",dtviagem:"2015-12-31"}
```

- O JSON pode ou não ter aspas duplas nos nomes dos campos que podem ser maiúsculos ou minúsculos. Já os valores precisam ter aspas duplas, podendo não ter apenas em caso do valor ser número (int ou float).
- Apenas 1 JSON por xObs, o primeiro. Outros serão ignorados.
- Informações de parâmetros da obs.html no cabeçalho do código fonte da página obs.html.

2. Tags do Grupo ObsCont

Caso o Segurado prefira integrar em seu sistema TMS a geração automática dos Parâmetros Adicionais, estes devem ser inseridos no XML como tags adicionais do grupo ObsCont (ou ObsFisco) em compl.

ATENÇÃO: Caso existam parâmetros no xObs, todos os ObsCont serão ignorados.

Página 159 do manual do CT-e v3.00.

Página 225 do manual do NF-e v6.00.

Os atributos **xCampo** do XML levam os mesmos nomes dos campos adicionais que vão nas observações (acima), assim como as tags **xTexto** aceitam os valores no mesmo formato.

Exemplo de utilização no CT-e:

```
<compl>
  <ObsCont xCampo="dtviagem">
    <xTexto>2015-12-31</xTexto>
  </ObsCont>
  <ObsCont xCampo="placa">
    <xTexto>ABC1234</xTexto>
  </ObsCont>
</compl>
```

3. Tabela de parâmetros

Descrição	Nome xcampo	Caracteres	Ocorrência	Informações
Sem valor declarado de transporte aéreo	semvalaereo	1	*: 0-1	s- sim; n- não;
Operação de ocd	ocd	1	*: 0-1	1- OCD; 2- OCDE; 3- OCD Especial; 4- OCDE Especial;
Operação de redespacho - Início	cmunini	1-n	*: 0-1	Código IBGE de municípios
Operação de redespacho - Fim	cmunfim	1-n	*: 0-1	Código IBGE de municípios
Unidade Federativa da fronteira	fronteira	2	*: 0-1	UF - 2 letras
Valor de benefícios internos	vbenefint	1-11	*: 0-1	Ponto decimal (.)
Número da viagem	viagem	1-9	*: 0-1	Número inteiro
Placa do veículo	placa	7	*: 0-1	ABC1234 (3 letras e 4 números)
Data da viagem	dtviagem	10	*: 0-1	aaaa-mm-dd
S. Embarcador	embarcador	1	Transp: 0-1	s- sim; n- não;
Transporte de mudanças	mudanca	1	Transp: 0-1	s- sim; n- não;
Operação de subcontratação	tpserv	1	Transp: 0-1	Número inteiro (Vide manual CT-e Sefaz)
Valor de vasilhame	vvasilhame	1-11	Transp: 0-1	Ponto decimal (.)
Valor de container	vcontainer	1-11	Transp: 0-1	Ponto decimal (.)
Cobertura de apólice	cobertura	3	Emb: 0-1	Número inteiro
Coberturas adicionais	adicionais	3-n	Emb: 0-1	Alfa-numéricos separados por vírgula (,)
Cobre NF de entrada	cobnfentrada	1	Emb: 0-1	s- sim; n- não;
Valor do lucro esperado	lucro	1-11	Emb: 0-1	Ponto decimal (.)
Valor de beneficiamento	vbenef	1-11	Emb: 0-1	Ponto decimal (.)
Valor de despesas	vdespesas	1-11	Emb: 0-1	Ponto decimal (.)
Modalidade do frete	modfrete	1	Emb: 0-1	Número inteiro

Acesso via Postman

- Download do aplicativo para PC: <https://www.getpostman.com/apps>
- Documentação Postman da API AVerbePorto (**recomendado**):
 1. Descrição da API: <https://documenter.getpostman.com/view/207913/RWgozeRg>
Execute clicando no botão laranja “Run in Postman” ou
 2. **Collection:** [Importe esta Collection pronta](#) (Import > Link ou File) no aplicativo (Vídeo: <https://youtu.be/M-qHvBhULes>)
 3. Na aba Collection do aplicativo, irá aparecer uma nova coleção chamada “www.AverbePorto.com.br”
- Gerar **exemplos** para outras linguagens na aplicação:
 1. Clicar em “Code” (abaixo dos botões Send e Save)
 2. Selecionar a linguagem no combo localizado no topo da janela “Generate Code Snippets”

- Passos para configuração manual (* é recomendado utilizar a collection pronta acima):

- URL

POST: <https://apis.averbeporto.com.br/php/conn.php>

- Mais sobre a URL em [Manual de Integração - Averbepo](#)

- Login

Passar os parâmetros de login para efetuá-lo e receber o [Cookie](#) de Sessão. O Postman utilizará o cookie automaticamente para a mesma URL.

- [Não utilize a senha comum para acesso web do usuário para acessar a API](#). Acessando o [sistema Web](#), vá no módulo **Cadastro do Usuário** e gere uma senha de API própria, clicando no **X** para iniciar o processo.

Body (x-www-form-urlencoded):

mod: login

comp: 5

user: SEU_USUARIO

pass: SUA_SENHA

- Envio de arquivo

Upload do arquivo XML ou ZIP.

Body (form-data):

mod: Upload

comp: 5

path: guarda/php/

file: "Selecionar arquivo xml ou zip"

Headers:

"Deixar completamente vazio" (inclusive remover o Content-type)

Exemplo em PHP (cURL)

```
<?php
```

```
/**
```

```
* Open an url on https using curl and return content
```

```
* @param string url      The url to open
```

```
* @param string refer    Referer (optional)
```

```
* @param mixed usecookie If true, cookie.txt will be used as default, or the usecookie value.
```

```
* @return string
```

```
*/
```

```
function open_https_url($url,$refer = "", $usecookie = false) {
```

```
    if ($usecookie) {
```

```
        if (file_exists($usecookie)) {
```

```
            if (!is_writable($usecookie)) {
```

```
                return "Can't write to $usecookie cookie file, change file permission to 777 or remove read only for windows.";
```

```
            }
```

```
        } else {
```

```
            $usecookie = ($usecookie === true)? "cookie.txt" : $usecookie;
```

```
        if (!touch($usecookie)) {
```

```

        return "Can't write to $usecookie cookie file, change file permission to 777 or remove read only for
windows.";
    }
}
}

```

```

$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, $url);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, 2);
curl_setopt($ch, CURLOPT_HEADER, 0);
curl_setopt($ch, CURLOPT_USERAGENT, "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)");

```

```

if ($usecookie) {
    curl_setopt($ch, CURLOPT_COOKIEJAR, $usecookie);

    curl_setopt($ch, CURLOPT_COOKIEFILE, $usecookie);
}

```

```

if ($refer != "") {

    curl_setopt($ch, CURLOPT_REFERER, $refer );

}

```

```

curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
return $ch;
}

```

```

/**
 * Limpeza ao terminar de executar
 * Requer $ws
 */

```

```

function ws_shutdown(){
    global $ws;

    if (file_exists($ws['cookie'])) {
        unlink($ws['cookie']);
    }
}
register_shutdown_function('ws_shutdown');

```

```

/**
 * Ajax Request

```

```

 * Requer $ws Global Config (comp/path/cookie)
 * aPost (array) json params
 * sModule (string) mod (i.e. login/Upload/Retrieve)
 * $sConn (string) URI to connect
 */

```

```

function websysRequest($aPost, $sModule = 'login', $sConn = 'https://apis.averbeporto.com.br/php/conn.php') {
    global $ws;

```

```

if (!isset($aPost['comp'])) { $aPost['comp'] = $ws['comp']; }
if (!isset($aPost['path'])) { $aPost['path'] = $ws['path']; } elseif ($aPost['path'] == '') {
    unset($aPost['path']); }
$aPost['mod'] = $sModule;

$ch = open_https_url($sConn, '', $ws['cookie']);

curl_setopt($ch, CURLOPT_POST, 1);
curl_setopt($ch, CURLOPT_POSTFIELDS, $aPost);

$resp = curl_exec($ch);
curl_close($ch);

return $resp;
}
?>

```

Exemplo de Uso:

```

<?php
require_once('func.php');

// Exemplo Config $ws
$ws = array(
    'comp' => 5,
    'path' => 'guarda/php/',
    'conn' => 'https://apis.averbeporto.com.br/php/conn.php',
    'cookie' => tempnam(sys_get_temp_dir(), 'ws_'),
    'logged' => ''
);

/**
 * Envia arquivo
 *
 * @param {String} Conteudo do arquivo
 * @param {Array} Usuario e senha. Ex.: array('user'=>'USUARIO', 'pass'=>'SENHA', 'path'=>")
 * @param {String} (optional) Remetente (em caso de email)
 * @return {Array} Retorna resposta do webservice
 */
function sendFile($sFileContent, $aUser, $sRecipient = ""){
    global $ws;
    $file = tmpfile();
    fwrite($file, $sFileContent);
    rewind($file);
    $meta = stream_get_meta_data($file);
    $mime = mime_content_type($meta['uri']);

    $post = array(
        'file' => (version_compare(PHP_VERSION, '5.5') >= 0)? new CURLFile($meta['uri'], $mime) :
        '@'.$meta['uri'].';type='.$mime
    );

```

```

if ($sRecipient) {
    $post['recipient'] = $sRecipient;
}

// Login
if ($ws['logged'] != $aUser['user']) {
    $res = json_decode(websysRequest($aUser), true);
    if (isset($res['logout']) && $res['logout']) {
        //ws_log('MAIL2EG: ['.aUser['user'].']: '.posix_getpid().': Falha do login. ');
    }
} else {
    $res['success'] = $res['C'] = true;
}

// Upload
if ($res['success'] && isset($res['C'])) {
    $ws['logged'] = $aUser['user'];
    $res = json_decode(websysRequest($post, 'Upload'), true);
}

fclose($file);
return $res;
}

$aUser = array(
    'user' => 'USUARIO',
    'pass' => 'SENHA',
    'path' => "
);
$sFileContent = file_get_contents('ARQUIVO.xml');

$res = sendFile($sFileContent, $aUser);

print_r($res);
?>

```

Exemplo em Delphi (Indy)

```

unit PortoIndy;
{
    Créditos:
    Francisco Caffagni
    O2 Tecnologia - (30/08/2016)

    Acesso ao Rest da Porto Seguro

    URL:
    https://apis.averbeporto.com.br/php/conn.php
}

```

interface

uses

System.SysUtils, System.Variants, System.Classes,
IdHTTP, IdMultipartFormData,
IdCookieManager, IdURI,
Data.DBXJSON;

type

TPortoSeguroIndy = **class**

private

FPostCookieStream: TIdMultiPartFormDataStream;
FPostFileStream: TIdMultiPartFormDataStream;
FRespCookieStream: TStringStream;
FRespFileStream: TStringStream;
FIdHTTP: TIdHTTP;
FCookie: **string**;
FCookieName: **string**;
FFileName: **string**;
FMensagemRetorno: **string**;
FPassword: **string**;
FResponse: **string**;
FResponseCookie: TJSONValue;
FResponseCookieStr: **string**;
FResponseFile: TJSONValue;
FResponseFileStr: **string**;
FSucesso: **boolean**;
FCookieText: **string**;
FURL: **string**;
FUsername: **string**;

procedure SetPassword(const Value: **string**);

procedure SetURL(const Value: **string**);

procedure SetUsername(const Value: **string**);

public

constructor Create;

destructor Destroy; **override**;

function Upload(AFileName: **string**): **boolean**;

property Cookie: **string** **read** FCookie;

property CookieName: **string** **read** FCookieName;

property CookieText: **string** **read** FCookieText;

property MensagemRetorno: **string** **read** FMensagemRetorno;

property Password: **string** **read** FPassword **write** SetPassword;

property Response: **string** **read** FResponse;

property ResponseCookie: TJSONValue **read** FResponseCookie;

property ResponseCookieStr: **string** **read** FResponseCookieStr;

property ResponseFile: TJSONValue **read** FResponseFile;

property ResponseFileStr: **string** **read** FResponseFileStr;

property Sucesso: **boolean** **read** FSucesso;

property URL: **string** **read** FURL **write** SetURL;

property Username: **string** **read** FUsername **write** SetUsername;

end;

const

AURI: **string** = 'https://apis.averbeporto.com.br';

implementation

```
{ TPortoSeguroIndy }
{$REGION 'Constructor / destructor methods'}
constructor TPortoSeguroIndy.Create;
begin
  //FUsername := 'USUARIO';
  //FPassword := 'SENHA';

  FPostCookieStream := TIdMultiPartFormDataStream.Create;
  FPostCookieStream.AddFormField('mod', 'login');
  FPostCookieStream.AddFormField('comp', '5');

  FPostFileStream := TIdMultiPartFormDataStream.Create;
  FPostFileStream.AddFormField('mod', 'Upload');
  FPostFileStream.AddFormField('comp', '5');
  FPostFileStream.AddFormField('path', 'guarda/php/');
  FPostFileStream.AddFormField('recipient', 'T'); // Transportador

  FRespCookieStream := TStringStream.Create("");
  FRespFileStream := TStringStream.Create("");

  FIdHTTP := TIdHTTP.Create(nil);
  FIdHTTP.Request.ContentType := FPostCookieStream.RequestContentType;
  FIdHTTP.Request.UserAgent := 'Mozilla/5.0 (Windows NT 6.1; WOW64; rv:12.0) Gecko/20100101 Firefox/12.0';

  FCookie := "";
  FCookieName := "";
  FResponseFile := TJSONValue.Create;
  FResponseCookie := TJSONValue.Create;
  FCookieText := "";
  FURL := AURI + '/php/conn.php';
  FSucesso := false;
end;

destructor TPortoSeguroIndy.Destroy;
begin
  FPostCookieStream.Free;
  FPostFileStream.Free;
  FRespCookieStream.Free;
  FRespFileStream.Free;
  FIdHTTP := nil;
inherited;
end;
{$ENDREGION}

{$REGION 'Class methods'}
function TPortoSeguroIndy.Upload(AFileName: string): boolean;
var
  URI: TIdURI;
begin
  try
    if (FUsername = "") then
      raise Exception.Create('User Porto Seguro não pode estar em branco.');
```

```

    raise Exception.Create('Password Porto Seguro não pode estar em branco. ');
if (AFileName = "") then
    raise Exception.Create('Arquivo: ' + AFileName + ' não pode estar em branco. ');
if (not FileExists(AFileName)) then
    raise Exception.Create('Arquivo: ' + AFileName + ' não existe. ');

FFilename := AFileName;
// Get Cookie
FPostCookieStream.AddFormField('user', FUsername);
FPostCookieStream.AddFormField('pass', FPassword);
FIdHTTP.Post(FURL, FPostCookieStream, FRespCookieStream);

FResponseCookieStr := FRespCookieStream.DataString;
FResponseCookie := TJSONString.Create(FResponseCookieStr);
FCookieName := FIdHTTP.CookieManager.CookieCollection.Cookies[0].CookieName;
FCookieText := FIdHTTP.CookieManager.CookieCollection.Cookies[0].CookieText;
FCookie := Copy(FCookieText, Pos('portal[ses]', FCookieText), Pos(';', FCookieText)-1);

// Upload file
FPostFileStream.AddFile('file', FFileName);

FIdHTTP.AllowCookies := true;
FIdHTTP.CookieManager := TIdCookieManager.Create;
URI := TIdURI.Create(AURI);

FIdHTTP.CookieManager.AddServerCookie(FCookie, URI);
FIdHTTP.Request.ContentType:= 'multipart/form-data';
FIdHTTP.Post(FURL, FPostFileStream, FRespFileStream);
FResponseFileStr := FRespFileStream.DataString;
FResponse := FRespFileStream.DataString;
FResponseFile := TJSONString.Create(FResponse);

{
1. Json de retorno:
(
[success] => 1
[S] => Array
(
[P] => 1 // Processado (xml guardado com sucesso)
[D] => 0 // Duplicado (xml préexistente)
[R] => 0 // Rejeitado (xml não parece ser do tipo certo)
[N] => 0 // Negado (Não é xml ou zip)
)
)
}

//{"success":1,"S":{"P":0,"D":0,"R":0,"N":1}}
FSucesso := ((Pos("success":1', FResponse) > 0) and (Pos("P":1', FResponse) > 0));

if (Pos("P":1', FResponse) > 0) then
    FMensagemRetorno := 'Processado (xml guardado com sucesso)'
else if (Pos("D":1', FResponse) > 0) then
    FMensagemRetorno := 'Duplicado (xml préexistente)'
else if (Pos("R":1', FResponse) > 0) then
    FMensagemRetorno := 'xml não parece ser do tipo certo'
else if (Pos("N":1', FResponse) > 0) then

```

```

    FMensagemRetorno := 'Negado (Não é xml ou zip)'
else
    FMensagemRetorno := 'Erro inesperado: ' + FResponse + '. Entre em contato com a operadora de seguro.';

    Result := FSucesso;

except
    on e: Exception do
        raise Exception.Create('[PortoIndy.TPortoSeguroIndy.Upload]: ' + e.Message);
    end;
end;
{$SENDREGION}

{$REGION 'getter / setter methods'}
procedure TPortoSeguroIndy.SetPassword(const Value: string);
begin
    FPassword := Value;
end;

procedure TPortoSeguroIndy.SetURL(const Value: string);
begin
    FURL := Value;
end;

procedure TPortoSeguroIndy.SetUsername(const Value: string);
begin
    FUsername := Value;
end;
{$SENDREGION}

end.

```