



API RESTful Multicanal

DOCUMENTATION

Sommaire

Table des matières

Introduction.....	2
Authentification	2
Appels	2
Fonctions disponibles	3
Types de requêtes	3
Types de réponses	4
Exemple GET.....	5
Exemple POST	6
Contenu de la requête.....	7
Champs obligatoires	8
Champs communs.....	8
Champs spécifiques	9
Exemples de contenu pour les requêtes POST	10
Exemple de pièces jointes	10
Exemple d'image embedded.....	10
Exemple de campagne	11
Envoi via un modèle et une liste	11
Envoi en passant tous les paramètres	11
Exemple de courrier.....	12
Requête POST	12
Exemple de fichier	13
Charger un fichier post.....	13
Interroger le stockage de fichier (PUT)	13
EMAIL – CURL	16
EMAIL – C# – RestCharp	17
EMAIL – Java – Unirest	17
EMAIL – JavaScript – JQuery	18
EMAIL – PHP – http_Request2.....	19

SMS – Curl.....	20
SMS – C# - RestSharp.....	20
SMS – Java – Unirest	21
SMS – JavaScript – JQuery.....	21
SMS – PHP – http_Request2.....	22
Configurer DKIM & SPF	23

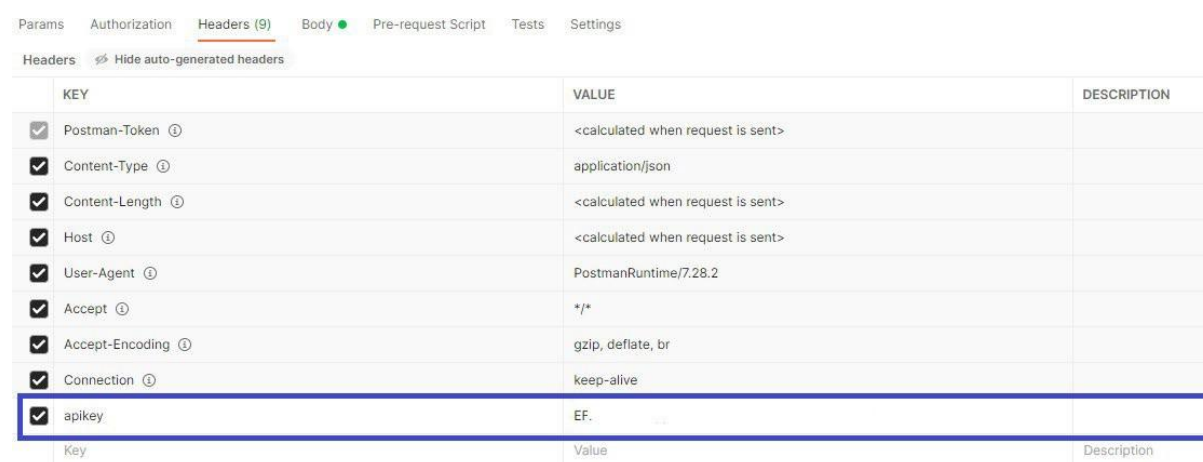
Introduction

La documentation API RESTful ci-après est un complément de la documentation complète des fonctions API à retrouver en suivant l'url suivante :

<https://extranet.efidem.com/api/swagger/index.html>

Authentification

L'authentification se fait via un header dénommé « apikey » dont la valeur est la clé d'API fournie par EFIDEM.



KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> Postman-Token ⓘ	<calculated when request is sent>	
<input checked="" type="checkbox"/> Content-Type ⓘ	application/json	
<input checked="" type="checkbox"/> Content-Length ⓘ	<calculated when request is sent>	
<input checked="" type="checkbox"/> Host ⓘ	<calculated when request is sent>	
<input checked="" type="checkbox"/> User-Agent ⓘ	PostmanRuntime/7.28.2	
<input checked="" type="checkbox"/> Accept ⓘ	*/*	
<input checked="" type="checkbox"/> Accept-Encoding ⓘ	gzip, deflate, br	
<input checked="" type="checkbox"/> Connection ⓘ	keep-alive	
<input checked="" type="checkbox"/> apikey	EF.	

Des droits sont appliqués sur chaque « apikey » selon les fonctions disponibles.

Appels

Le nombre d'appels est limité à 2000 appels simultanés toutes les 10 secondes. Au-delà de ce ratio, un code 429 est renvoyé.

Fonctions disponibles

Les fonctions disponibles sont les suivantes :

Nom de la route	Description
Blacklist	Gestion de la liste noire
Campaign	Fonctions de création et de suivi de campagnes
Dests	Gestion des destinataires
Domains	Gestion des domaines d'expédition (pour les emails)
Email	Envoi d'email
Files	Gestion des fichiers
Letters	Envoi de courriers
Lists	Gestion des listes de destinataires
MO	Gestion des SMS entrants
OTP	Gestion des messages OTP
SMS	Envoi de messages courts
Tools	Différents outils (compteur de caractères SMS...)
Users	Gestion des utilisateurs

Types de requêtes

L'API dispose de plusieurs types de requêtes HTTP :

Type	Commentaire	Exemple
GET	Utilisé pour récupérer un objet sur une donnée spécifique	<code>/api/v1/sms/{idmessage}</code>
POST	Utilisé pour poster un nouvel objet sur le serveur sur la route spécifiée	<code>/api/v1/sms</code>
PUT	Utilisé pour une mise à jour d'une information déjà existante	<code>/api/v1/modeles</code>

Types de réponses

Il existe deux types de réponses possibles selon le statut http retourné par le webservice.

Code HTTP	Commentaire	Type de réponse
200	Utilisé pour passer en paramètre un identifiant de message client	<pre>{ "isOk": true, "message": "aaaaaaaa-0d86-4072-ae0b-1349d8fac029", "data": { "destination": "0102030405", "texte": "Test ", "oadc": "", } }</pre>
400	Votre requête n'est pas conforme	<pre>{ "error": "Message d'erreur ici" }</pre>
401	Fonction non autorisée. Potentiellement, votre apikey n'a pas le droit d'accéder à la fonction appelée.	
404	La fonction demandée n'existe pas à l'adresse appelée.	
405	La fonction demandée n'existe pas sur la route demandée	
500	Erreur de traitement côté serveur.	<pre>{ "StatusCode": 500, "ErrorMessage": "Erreur XYZ" }</pre>

Dans le cas du code Success (code 200), la réponse est décomposée comme suit selon le type de requête :

Exemple GET

```
{
  "isOk": true,
  "message": null,
  "data": {
    "requete_Valide": true,
    "statut_Description": null,
    "billingCode": null,
    "customerReference": null,
    "submissionDate": "2021-07-11T11:54:37.797",
    "statuses": [
      {
        "destination": "test@efidem.com",
        "recipient_NAME": "TEST",
        "description": "Ouvert",
        "response": "",
        "date_STATUT": "2021-07-11T11:57:00",
        "type_DESTINATION": "To"
      }
    ]
  }
}
```

Nom du champ	Commentaire	Type
isOk	Valeur booléenne indiquant si la requête a été positive ou négative. Dans le cas de la valeur « true », le champ message contiendra l'identifiant unique de la requête. Dans le cas contraire, le champ message contiendra le descriptif de l'erreur rencontrée.	Booléen
message	Valeur en corrélation avec le champ isOk. Si is Ok est à « true », le champ message contiendra un id de type Guid. Dans le cas contraire, ce champ contiendra un message texte expliquant.	Texte
data	La requête initiale que vous avez envoyée	T

Exemple POST

Média	Contenu
SMS	<pre>{ "isOk": true, "message": "aaaaaaaa-7fc1-414f-bde8-2f7759b08f6b", "data": { ... } }</pre>
EMAIL	<pre>{ "isOk": true, "message": "aaaaaaaa-7fc1-414f-bde8-2f7759b08f6b", "data": { ... } }</pre>

La section « data » reprend le contenu de la requête ayant initié la demande.

Nom du champ	Commentaire	Type
isOk	Valeur booléenne indiquant si la requête a été positive ou négative. Dans le cas de la valeur « true », le champ message contiendra l'identifiant unique de la requête. Dans le cas contraire, le champ message contiendra le descriptif de l'erreur rencontrée.	Booléen
message	Valeur en corrélation avec le champ isOk. Si isOk est à « true », le champ message contiendra un id de type Guid. Dans le cas contraire, ce champ contiendra un message texte expliquant.	Texte
data	La requête initiale que vous avez envoyée	T

Contenu de la requête

Le contenu des requêtes POST est en JSON. Voici un exemple pour l'envoi de SMS.



Le header « Content-Type » doit être sur « `application/json` ».

EMAIL
<pre> { "fromAddress": { "name": "TEST", "address": "dev@efidem.com" }, "toAddresses": [{ "name": "TEST", "address": "ms@efidem.com" }], "ccAddresses": [], "replyToAddress": { "name": "TEST", "address": "noreply@efidem.com" }, "attachements": [], "subject": "test efidem subject", "content": "test efidem content
 test html
", "trackOpen": true, "trackClicks": true } </pre>
SMS
<pre> { "Destination": "0601020304", "Texte": "Test de SMS à partir de api REST", "OADC": "" } </pre>

Champs obligatoires

Dans la section « Schémas » du Swagger, vous retrouvez les différents schémas des données pour les différentes fonctions. Les champs dit obligatoires sont indiqués avec une étoile rouge. Exemple ci-dessous avec un message SMS :

```

SMSMessage {
  destination* string
  texte* string
  oadc string
  billingCode string
  customerReference string
  attention1 string
  nullable: true
}

```

Champs communs

Il existe différents champs communs pour l'ensemble des médias.

Nom du champ	Commentaire	Longueur max
ReferenceClient	Utilisé pour passer en paramètre un identifiant de message client	200
BillingCode	Code de facturation permettant d'aider à la ventilation des factures (Business Unit, Code de service, etc...)	200
attentionX	Divers champs d'attention pour être utilisé entant que champs d'informations ou champs variables via le texte @ATTNx@ ou x est le numéro du champ d'attention (de 1 à 8)	200

Champs spécifiques

Voici quelques champs spécifiques par média :

Média	Nom du champ	Description
SMS	OADC	Emetteur du message SMS. Longueur de 11 caractères alphanumériques
EMAIL	ATTACHMENTS :CONTENT	Contenu du fichier en base64
EMAIL	ATTACHMENTS :ISINLINE	Booléen indiquant si un fichier (généralement une image) doit être inclus dans le contenu d'un email
EMAIL	ATTACHMENTS :CONTENTID	Important dans le cas d'une pièce jointe ayant le paramètre ISINLINE à true. Il s'agit ici du nom présence dans la source de la balise IMG
EMAIL	ATTACHMENTS :TRACKOPEN	Suivre les ouvertures de mails
EMAIL	ATTACHMENTS :TRACKCLICS	Suivre les clics d'un email
FILES	FILENAME	Nom du fichier
FILES	FILECONTENT	Contenu du fichier en base64
Letters	PDFCONTENT	Contenu du fichier PDF en base64
Letters	ENVELOPETYPE	1=C6, 2=C5
Letters	ENVELOPEWINDOWTYPES	SMPL (simple fenêtre), DBL (double)
Letters	DONOTSEND	Booléen pour test du webservice permettant de vérifier la prise en compte d'un courrier sans l'envoyer

Voici quelques champs spécifiques par route :

Route	Nom du champ	Description
Campaign	TYPEMEDIA	1 = SMS 2 = EMAIL
Files	ACTIONNAME	0 = Liste tous les fichiers 1 = Donne les informations d'un fichier 2 = Suppression d'un fichier

Exemples de contenu pour les requêtes POST

Exemple de pièces jointes

```
{
  "fileName": "Fichier.pdf",
  "content": "JVBERi0xLjcNCiWg==",
  "contentID": null,
  "contentType": "application/pdf",
  "isInline": false
},
```

Exemple d'image embedded

```
{
  "fileName": "iconXYZ",
  "content": "iVBORw0KGgoAAAANSUhEUgAAAg==",
  "contentID": "iconXYZ",
  "contentType": "image/png",
  "isInline": true
},
```

Et dans le contenu du mail, la balise IMG doit correspondre aux champs ci-dessous :

```
<img height=21 style=\"max-width:100%;\" src=\"cid:iconXYZ\" alt=\"Mon image\"/>
```

Exemple de campagne

Pour une campagne, deux modes sont possibles :

- A/ Envoi via l'utilisation d'un modèle de message
- B/ En passant tous les paramètres dans la requête

A noter qu'en utilisant l'option A, il faut au préalable créer un modèle via notre site et renseigner les différents champs (sujet, champ émetteur pour l'email par exemple)

Envoi via un modèle et une liste

Dans l'exemple ci-dessous, une liste de contact est appelée via le tableau contactsLists et les paramètres pour l'envoi (contenu, emetteur...) sont récupérés via un id de modèle.

```
{
  "campaignName": "TEST campagne API",
  "billingCode": "API",
  "typeMedia": 1,
  "contactsLists": ["c89fe38a-0000-zzzz-aaaa-e11eda32e349"],
  "recipientsOnTheFly": ["0607011689"],
  "idModeleMessage": "a0a5d565-0000-zzzz-aaaa-fff9f1f7beb6",
  "checkDuplicates": true
}
```

Envoi en passant tous les paramètres

Dans l'exemple ci-dessous, tous les paramètres de la campagne sont passés dans la requête.

```
{
  "campaignName": "TEST campagne API Email",
  "billingCode": "API",
  "typeMedia": 2,
  "contactsLists": [],
  "recipientsOnTheFly": ["janedoe@efd.com"],
  "content": "<h1>test API Rest EMAIL Campagne</h1><h6>From PostMan</h6>",
  "Subject": "Sujet de l'email",
  "fromname": "John Doe",
  "frommail": "info@efidem.com",
  "checkDuplicates": true
}
```

Exemple de courrier

Requête POST

Voici un détail des différentes énumérations nécessaires à l'envoi d'un courrier postal.

EnvelopeTypes

Valeur	Code Enum
C6	1
C5	2

PostageTypes

Valeur	Code Enum
ECOPLI_GRAND_COMPTE	0
RECOMMANDE	1
RECOMMANDE_AR	2
LG2	3

EnvelopeWindowTypes

Valeur	Code Enum
SMPL	0
DBL	1

Exemple de fichier

Charger un fichier post

Si le sous-dossier mentionné dans subDirectory pour l'utilisateur sélectionné n'existe pas, il est alors créé par le système.

```
{
  "userName": "monutilisateur@client.com",
  "subDirectory": null,
  "fileName": "TEST_EFIDEM.pdf",
  "fileContent": "JVBERi0xRU9G..."
}
```

Interroger le stockage de fichier (PUT)

Via une commande PUT, il est possible d'exécuter diverses actions. Le champ actionName sert à définir l'action à réaliser. Il peut prendre 3 valeurs :

Valeur	Correspondance
0	Récupérer les informations de tous les fichiers
1	Récupérer les informations d'un fichier
2	Supprimer un fichier

Exemple de requête

<pre>{ "actionName": 0, "folder": "pdf", "file": "", "userName": "test@test.test" }</pre>	<p>Obligatoire Correspond à l'action à exécuter (0,1,2)</p> <p>Correspond au sous-dossier (optionnel)</p> <p>Nom du fichier (pour actions 1 et 2)</p> <p>Obligatoire Nom d'utilisateur</p>
---	--

La requête ci-dessus liste donc tous les fichiers pour l'utilisateur test@test.test au niveau du sous-répertoire « pdf ».

D'autres exemples de requêtes :

Affichage des informations d'un fichier en particulier :

<pre>{ "actionName": 1, "folder": "pdf", "file": "test.pdf", "userName": "test@test.test" }</pre>	<p>Obligatoire Affichage des informations d'un fichier</p> <p>Correspond au sous-dossier (optionnel)</p> <p>Obligatoire Nom du fichier</p> <p>Obligatoire Nom d'utilisateur</p>
---	---

Suppression d'un fichier :

<pre>{ "actionName": 2, "folder": "pdf", "file": "test.pdf", "userName": "test@test.test" }</pre>	<p>Obligatoire Suppression d'un fichier</p> <p>Correspond au sous-dossier (optionnel)</p> <p>Obligatoire Nom du fichier</p> <p>Obligatoire Nom d'utilisateur</p>
---	--

Exemple de réponse :

<pre>{ "fileName": "fichier.txt", "directory": "test@test.test", "extension": ".txt", "creationTime": "2022-09-22T14:37:22.6930569+02:00", "size": 345, "url": "https://url.ef./f?f=JsjsbOz4EHVyh1q4xX2fljzeAjk", "content": null },</pre>	<p>Nom du fichier</p> <p>Répertoire</p> <p>Extension</p> <p>Date de création sur le système</p> <p>Taille en octets</p> <p>Url d'accès publique</p>
--	---

Annexes

Exemples de code

EMAIL – CURL

```
curl --location --request POST
'https://extranet.efidem.com/api/v1/email' \
--header 'ApiKey: EF.xxxxxxxxxxxxxxxxxxxx' \
--header 'Content-Type: application/json' \
--data-raw '{
  "fromAddress"
  : {
    "name": "TEST",
    "address": "dev@efidem.com"
  },
  "toAddresses": [
    {
      "name": "TEST",
      "address": "technique@efidem.com"
    }
  ],
  "ccAddresses":
  [],
  "replyToAddress"
  : {
    "name": "TEST",
    "address": "noreply@efidem.com"
  },
  "attachements": [],
  "subject": "test efidem subject",
  "content": "test efidem content<br/> test
html<br/>", "trackOpen": true,
  "trackClicks": true
}'
```

EMAIL – C# – RestCharp

```
var client = new
RestClient("https://extranet.efidem.com/api/v1/email");
client.Timeout = -1;
var request = new RestRequest(Method.POST);
request.AddHeader("ApiKey",
"EF.xxxxxxxxxxxxxxxxxxxxxx");
request.AddHeader("Content-Type",
"application/json"); var body = @"{
" + "\n" +
@"
""fromAddress"":
{ " + "\n" +
@"    ""name"":
""TEST"", " + "\n" +
@"    ""address"":
""dev@efidem.com"" " + "\n" +
@" },
" + "\n" +
@"
""toAddresses"":
[ " + "\n" +
@"    {
" + "\n" +
@"        ""name"":
""TEST"", " + "\n" +
@"        ""address"":
""technique@efidem.com"" " + "\n" +
@"    }
" + "\n" +
@" ],
" + "\n" +
@"
""ccAddresses"": [], " + "\n" +
@"
""replyToAddress"" : { " + "\n" + @" ""name"": ""TEST"", " + "\n" +
@"
""address"": ""noreply@efidem.com""
" + "\n" +
@" },
" + "\n" +
@"
""attachements"": [], " + "\n" +
@"
""subject"": ""test efidem subject"", " + "\n" +
@"
""content"": ""test efidem content<br/> test html<br/>"" , " + "\n" +
@"
""trackOpen"": true, " + "\n" +
@"
""trackClicks"":
true " + "\n" +
@"}";

request.AddParameter("application/json", body,
ParameterType.RequestBody); IRestResponse response =
client.Execute(request); Console.WriteLine(response.Content);
```

EMAIL – Java – Unirest

```
Unirest.setTimeouts(0, 0);
HttpResponse<String> response = Unirest.post("https://extranet.efidem.com/api/v1/
email")
    .header("ApiKey", "EF.xxxxxxxxxxxxxxxxxx")
    .header("Content-Type", "application/json")
    .body("{\r\n  \"fromAddress\": {\r\n    \"name\": \"TEST\", \r\n    \"address\":
    \"dev@efidem.com\" \r\n  }, \r\n  \"toAddresses\": [\r\n    {\r\n      \"name\": \"
TEST\", \r\n      \"address\": \"technique@efidem.com\" \r\n    }, \r\n    \r\n  ], \r\n  \"
ccAddresses\": [], \r\n  \"replyToAddress\" : {\r\n    \"name\": \"TEST\", \r\n
    \"address\": \"noreply@efidem.com\" \r\n  }, \r\n  \"attachements\": [], \r\n
\r\n  \"subject\": \"test efidem
subject\", \r\n  \"content\": \"test efidem content<br/> test html<br/>\", \r\n
  \"trackOpen\": true, \r\n  \"trackClicks\": true\r\n}")
    .asString();
```

EMAIL – JavaScript – JQuery

```
var settings = {
  "url":
    "https://extranet.efidem.com/api/v1/email",
  "method": "POST",
  "timeout": 0,
  "headers": {
    "ApiKey":
      "EF.xxxxxxxxxxxxxxxxxxxxx",
    "Content-Type":
      "application/json"
  },
  "data": JSON.stringify({
    "fromAddress": {
      "name": "TEST",
      "address": "dev@efidem.com"
    },
    "toAddresses": [
      {
        "name": "TEST",
        "address": "technique@efidem.com"
      }
    ],
    "ccAddresses":
      [],
    "replyToAddress":
      {
        "name": "TEST",
        "address": "noreply@efidem.com"
      },
    "attachements": [],
    "subject": "test efidem subject",
    "content": "test efidem content<br/> test
html<br/>", "trackOpen": true,
    "trackClicks": true
  }),
};
```

EMAIL – PHP – http_Request2

```
<?php
require_once 'HTTP/Request2.php';
$request = new HTTP_Request2();
$request->setUrl('https://extranet.efidem.com/api/v1/email');
$request->setMethod(HTTP_Request2::METHOD_POST);
$request->setConfig(array( 'follow_redirects' => TRUE
));
$request->setHeader(array(
'ApiKey' => 'EF.xxxxxxxxxxxxxxxxxx', 'Content-Type' =>
'application/json'
));
$request->setBody('{
\n "fromAddress": {
\n   "name": "TEST",
\n   "address": "dev@efidem.com"
\n },
\n "toAddresses": [
\n   {
\n     "name": "TEST",
\n     "address": "technique@efidem.com"
\n   }
\n ],
\n   "ccAddresses": [],
\n   "replyToAddress" : {
\n     "name": "TEST",
\n     "address": "noreply@efidem.com"
\n   },
\n   "attachements": [],
\n "subject": "test efidem subject",
\n "content": "test efidem content<br/> test html<br/>",
\n "trackOpen": true,
\n "trackClicks": true
\n}');
try {
$response = $request->send();
if ($response->getStatus() == 200) {
echo $response->getBody();
}
else {
echo 'Unexpected HTTP status: ' . $response->getStatus() . ' ' .
$response->getReasonPhrase();
}
}
catch(HTTP_Request2_Exception $e) {
echo 'Error: ' . $e->getMessage();
}
}
```

SMS – Curl

```
curl --location --request POST
'https://extranet.efidem.com/api/v1/sms' \
--header 'ApiKey: EF.xxxxxxxxxxxxxxxxxx' \
--header 'Content-Type: application/json' \
--data-raw '{
    "Destination": "0601020304",
    "Texte": "Test de SMS à partir de
api REST", "OADC": ""
}'
```

SMS – C# - RestSharp

```
var client = new
RestClient("https://extranet.efidem.com/api/v1/sms");
client.Timeout = -1;
var request = new RestRequest(Method.POST);
request.AddHeader("ApiKey",
"EF.xxxxxxxxxxxxxxxxxx");
request.AddHeader("Content-Type",
"application/json"); var body = @"{
" + "\n" +
@"    ""Destination"":
""0601020304"", " + "\n" +
@"    ""Texte"": ""Test de SMS à partir de
api REST"", " + "\n" +
@"    ""OADC"":
"""" + "\n" +
@"}";
request.AddParameter("application/json", body,
ParameterType.RequestBody); IRestResponse response =
client.Execute(request); Console.WriteLine(response.Content);
```

SMS – Java – Unirest

```
Unirest.setTimeouts(0, 0);
HttpResponse<String> response =
Unirest.post("https://extranet.efidem.com/api/v1/sms")
    .header("ApiKey", "EF.xxxxxxxxxxxxxxxxxxxxxx")
    .header("Content-Type", "application/json")
    .body("{\r\n  \"Destination\": \"0601020304\", \r\n  \"Texte\": \"Test de SMS à partir de api REST\", \r\n  \"OADC\": \"\"");
```

SMS – JavaScript – JQuery

```
var settings = {
  "url":
    "https://extranet.efidem.com/api/v1/sms",
  "method": "POST",
  "timeout": 0,
  "headers": {
    "ApiKey":
      "EF.xxxxxxxxxxxxxxxxxxxxxx",
    "Content-Type":
      "application/json"
  },
  "data": JSON.stringify({
    "Destination":
      "0601020304",
    "Texte": "Test de SMS à partir de
      api REST", "OADC": ""
  })
};
```


SMS – PHP – http_Request2

```
<?php
require_once 'HTTP/Request2.php';

$request = new HTTP_Request2();
$request->setUrl('https://extranet.efidem.com/api/v1/sms');
$request->setMethod(HTTP_Request2::METHOD_POST);
$request->setConfig(array(
    'follow_redirects' => TRUE
));
$request->setHeader(array(
    'ApiKey' =>
    'EF.xxxxxxxxxxxxxxxxxxxx',
    'Content-Type' =>
    'application/json'
));
$request->setBody('{
\n  "Destination": "0601020304",
\n  "Texte": "Test de SMS à partir de api REST",
\n  "OADC": ""
\n}');
try {
    $response = $request->send();
    if ($response->getStatus() == 200) {
        echo $response->getBody();
    }
    else {
        echo 'Unexpected HTTP status: ' . $response->getStatus() . ' ' .
        $response->getReasonPhrase();
    }
}
catch(HTTP_Request2_Exception $e) {
    echo 'Error: ' . $e->getMessage();
}
```

Configurer DKIM & SPF

Pour mettre toutes les chances de votre côté et optimiser votre **délivrabilité EMAIL sans surcoût**, efidem recommande la mise en place d'un **DKIM & d'un SPF** pour votre domaine expéditeur.

Le but de ces protocoles est de **renforcer la sécurité** des emails envoyés via la **plateforme emailing efidem**.

Pourquoi mettre en place ces protocoles ?

Depuis la crise sanitaire liée à la covid-19, les fournisseurs d'accès à internet (FAI) ont été confrontés à une augmentation conséquente du volume de mail, des SPAMS et des tentatives de phishing. Pour mieux protéger leurs clients, les FAI ont renforcé leur sécurité.

Conséquence directe : Des emails légitimes qui arrivaient en boîte de réception sans paramétrages réseau spécifiques peuvent aujourd'hui être délivrés dans la boîte SPAM s'ils ne sont pas authentifiés avec un DKIM et un SPF.

Comment procéder ?

Pour authentifier vos emails, rendez-vous dans la **zone DNS** de votre domaine expéditeur, puis ajoutez une nouvelle entrée **CNAME** pour chaque ligne du tableau ci-dessous.

Si vous n'avez **pas accès** à l'administration de votre domaine, vous pouvez **transférer ce mail à votre service / prestataire informatique** qui pourra vous accompagner dans ces démarches.

Nom	Zone / Domaine	Type	Valeur / cible
Efd	votre-domaine.extension	CNAME	mta.efidem.com
efd1._domainkey	votre-domaine.extension	CNAME	key1.efidem.com
efd2._domainkey	votre-domaine.extension	CNAME	key2.efidem.com

Confirmez la création des enregistrements dans votre zone DNS par email.

Vous recevrez de la part de l'équipe support efidem un mail validant l'activation des protocoles.

En cas de difficultés ou de questions, n'hésitez pas à prendre contact avec l'équipe technique à l'adresse support@efidem.com ou par téléphone au [01 46 40 10 86](tel:0146401086).