Documentation de l'API HTTPS

Version 4.0.x

Libriciel SCOP

Document

Auteur	Eric Pommateau	***Date de diffusion***	18/02/2019
Chef de projet	Eric Pommateau	***N° de version***	4.0.x

Évolution du document

Version	Auteur	Nature des changements	Date
1.0	Eric Pommateau (Libriciel SCOP)	Documentation selon version 4.0.0	18/02/2019
2.0	Eric Pommateau (Libriciel SCOP)	Ajout de l'obligation de typer les pièces (actes)	21/03/2019

Licence

Ce document n'est pas libre de droits.

Ce manuel est publié sous la licence Creative Commons avec les particularités “Paternité – Partage à l'identique” (également connue sous l'acronyme CC BY-SA).

Détails de cette licence : http://creativecommons.org/licenses/by-sa/2.0/fr/

Libriciel SCOP

Table des matières

1. PRÉAMBULE

S²LOW (Service Sécurisé Libre inter-Opérable pour la Vérification et la Validation) est un logiciel de tiers de télétransmission multi-protocoles gérant les flux administratifs entre les collectivités territoriales et les administrations centrales. S²LOW implémente en premier lieu le protocole ACTES (flux de transmission aux préfectures pour le contrôle de légalité), d'autres protocoles devant venir s'ajouter progressivement (ex : HELIOS pour les transmission au Trésor Public).

L'usage de S²LOW se fait soit via une interface Web interactive soit à travers une API programme depuis un logiciel métier.

Ce document décrit l'API HTTPS qui est utilisée par l'interface Web pour le module ACTES. Cependant les applicatifs métier peuvent utiliser eux-mêmes ces appels par le biais de bibliothèques génériques implémentant le protocole HTTP.

S²LOW implémente le cahier des charges ACTES sans sur-couche de transport propriétaire.

2. CONNEXION À LA PLATE-FORME S²LOW

2.1. Explication

L'application désirant utiliser cette interface devra être capable d'établir une connexion TCP vers le port 443 du serveur Web exécutant l'application S²LOW. Elle devra ensuite effectuer une négociation SSL avec authentification mutuelle.

Pour cela elle devra présenter un certificat numérique reconnu par la plate-forme pour s'authentifier en tant qu'utilisateur attaché à une collectivité. Un compte utilisateur devra donc avoir été créé préalablement sur la plate-forme pour la détermination des droits d'accès et de l'appartenance à une collectivité. Le certificat doit avoir été importé sur le serveur S²LOW et doit donc être compatible avec la liste des certificats acceptés.

À partir de ce point, le dialogue HTTP peut avoir lieu au-dessus de la connexion SSL.

Des bibliothèques de programmation permettent d'effectuer ce travail fastidieux de connexion, par exemple Curl en C ou Jakarta HTTPClient en Java.

Si le certificat numérique correspond à plusieurs comptes, alors, ce certificat ne donne le droit d'utiliser qu'une seule fonction : la récupération de la liste des logins associés à ce certificat, sauf si les comptes associés utilisent un "certificat forward".

Afin de pouvoir se connecter à l'API, il convient alors d'envoyer le login et le mot de passe de l'utilisateur sous la forme d'une authentification HTTP en mode BASIC.

2.2. Récupération de la liste des logins

Nom du script : `/admin/users/api-list-login.php`

Méthode HTTP d'appel : GET ou POST

Description : Récupère la liste des logins associés au certificat présenté dans la requête.

Paramètres à fournir : aucun

Retour de l'appel :

- Rien si il n'y a pas de login associé ou bien si le certificat ne correspond pas.
- La liste des logins avec un login par ligne.

2.3. Identification par forward de certificat.

La version 2.2 de S²LOW introduit la notion de forward de certificat. Le principe est le suivant : un certificat d'authentification sert à la connexion HTTPS. Ce certificat correspond à plusieurs comptes utilisateurs (comme avec les comptes avec des logins/mots de passe). Mais chaque compte peut être identifié via un autre certificat numérique d'identification.

Les comptes configurés avec des certificats d'identification ne sont accessibles via l'API (il n'est pas possible de se connecter sur la console). Si un compte possède à la fois un certificat d'identification et un login/mot de passe, c'est le certificat d'identification qui prime sur le login : sa présentation est obligatoire et les logins/mots de passe ne sont pas vérifiés.

Afin de présenter un certificat d'identification, il faut disposer de sa partie publique dans le format DER, l'encoder en base 64 et l'envoyer avec la requête HTTP dans un champ d'en-tête personnalisé :

`org.s2low.forward-x509-identification`

Un exemple de connexion avec PHP et la bibliothèque Curl est disponible dans le code source de S2low `test/api/connect-forward-x509.php`

2.4. Fonction de test de connexion

Une fonction de l'API permet de tester la connexion sans faire d'action sur S2low :

Nom du script : `/api/test-connexion.php`

Méthode HTTP d'appel : GET

Droits nécessaires : utilisateur

Description : permet de valider la connexion à S2low

Paramètres à fournir : aucun

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par \n
- en cas d'échec : KO sur la première ligne terminée par \n, message d'erreur sur les lignes suivantes.

2.5. Fonction de récupération des informations de connexion

Cette fonction est disponible à partir de la version 3.0.10 de s2Low

Cette fonction de l'API permet de récupérer les informations sur l'utilisateur qui se connecte ainsi que sur sa collectivité de rattachement.

Nom du script : `/api/info-connexion.php`

Méthode HTTP d'appel : GET

Droits nécessaires : utilisateur

Description : permet de récupérer les informations de connexion

Paramètres à fournir : aucun

Retour de l'appel :

- en cas de succès de l'appel : un tableau json contenant les informations de connexion
- en cas d'échec : KO sur la première ligne terminée par \n, message d'erreur sur les lignes suivantes.

2.6. Fonction d'authentification par nonce

La version 2.6.0 de S2low lève une limitation sur les fonctions appelées directement par le client en redirection d'un portail tiers : l'impossibilité d'utilisation un certificat et un login/mot de passe.

Nom du script : `/api/get-nounce.php`

Méthode HTTP d'appel : **GET**

Droits nécessaires : utilisateur

Description : L'appel à cette fonction est fait avec un compte associé à une collectivité. Si l'authentification réussit un nonce est envoyé (valable 5 minutes). Ce nonce permettra de raccorder un autre utilisateur à la collectivité en passant le nonce en paramètre (et donc, sans passer par la mire de login/mot de passe). Ce nonce est principalement utilisé pour les api suivantes (appelé par le client) :

- Ordonner la télétransmission de l'acte (/actes_transac_post_confirm_api.php)
- Ordonner la télétransmission de plusieurs actes (/actes_transac_post_confirm_api.php)

Paramètres à fournir : aucun

Retour : un fichier json contenant le nounce sous la forme :

```
{"nounce":"contenu du nonce"}
```

Afin de s'authentifier par la suite avec le nonce, il convient de spécifier dans les paramètres GET les informations suivantes:

Nom paramètre	Description
nounce	le contenu du nonce retourné par get-nonce.php
login	le login ayant servi à la création du nonce
hash	le sha256 de la chaine "password:nounce" avec password, le mot de passe associé au login précédent et nounce, le nonce retourné par get-nonce.php

3. DESCRIPTION DE L'API HTTPS DE L'ADMINISTRATION DE S²LOW

3.1. Liste des groupes

Nom du script : `/admin/groups/admin_groups.php`

Méthode HTTP d'appel : **GET**

Droits nécessaires : Super Admin

Description : permet de lister les groupes de la plateforme.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`name`	facultatif	chaîne	Permet de filtrer la liste des groupes. Les noms des groupes récupérés contiennent la chaîne « name »

Retour :

une chaîne JSON contenant un tableau dont les lignes regroupent les informations sur les groupes.

Nom paramètre	Description
`id`	Identifiant unique du groupe
`name`	Nom du groupe
`status`	0 : Désactivé , 1 : Activé

3.2. Liste des types de collectivités

Nom du script : `/admin/authorities/admin_authority_types.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super admin ou administrateur de groupe

Description : permet de lister les types de collectivités

Paramètres à fournir : Aucun

Retour :

Une chaîne JSON contenant un tableau contenant des lignes de type id => description

Nom paramètre	Description
`id`	Identifiant interne du type de collectivité
`description`	Description du type de la collectivité

3.3. Liste des collectivités

Nom du script : `/admin/authorities/admin_authorities.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super Admin ou administrateur de groupe

Description : permet de lister les collectivités de la plateforme.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`name`	facultatif	chaîne	Permet de filtrer la liste des collectivités. Les noms des groupes récupérés contiennent la chaîne « name »
`type`	facultatif	chaîne	Filtre suivant un des identifiants de type de collectivité retourné par la fonction `admin_authority_types.php`
`groupe`	facultatif	chaîne	Filtre les collectivités qui appartiennent au groupe dont l'identifiant est donné par ce paramètre. Ce paramètre n'est utilisable que pour le super admin. Les identifiants de groupe peuvent être récupérés via la fonction `admin_groups.php`
`siren`	facultatif	Entier	Filtre sur tout ou partie d'un SIREN.

Retour :

une chaîne JSON contenant un tableau dont les lignes regroupent les informations sur les collectivités.

Nom paramètre	Description
`id`	Identifiant unique de la collectivité
`name`	Nom de la collectivité
`authority_group_id`	Identifiant du groupe de collectivité (voir la fonction `admin_groups.php`)

Autres paramètres de retour : `siren` , `address` , `postal_code` , `city` , `telephone` , `fax`

3.4. Liste des modules

Nom du script : `/admin/modules/admin_modules.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super admin ou administrateur de groupe

Description : permet de lister les modules actifs

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.

Retour :

Une chaîne JSON contenant un tableau contenant des lignes décrivant un module

Nom paramètre	Description
`id`	Identifiant interne du module
`description`	Description du module
`name`	Nom du module

3.5. Détail d'une collectivité

Nom du script : `/admin/authorities/admin_authority_detail.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super Admin, Admin de groupe, ou Admin de collectivité

Description : permet d'obtenir l'ensemble des informations sur une collectivité

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant de la collectivité

Retour :

une chaîne JSON contenant un tableau associatif contenant les informations sur la collectivité.

Nom paramètre	Description
`id`	Identifiant unique de la collectivité
`name`	Nom de la collectivité
`status`	0 :Désactivé , 1 : Activé
`authority_type_id`	Identifiant de type de collectivité retourné par la fonction admin_authority_types.php
`siren`	SIREN de la collectivité
`email`	Email de contact de la collectivité
`postal_code`	Code Postal
`city`	Ville
`department`	Département (sur trois chiffres)
`district`	Arrondissement (1 chiffre)
`telephone`	Téléphone
`fax`	Fax
`default_broadcast_email`	Email de notification sélectionné par défaut lors de la création d'un ACTE
`broadcast_email, address`	Email de notification non-sélectionné par défaut lors de la création d'un ACTE
`email_mail_securise`	Email de provenance des mails sécurisés (champ « from »)
`sae_wsdl`	URL du WSDL du SAE (Asal@e)
`sae_login`	Login du SAE
`sae_password`	Mot de passe du SAE
`sae_id_versant`	SEA Identifiant du service versant
`sae_id_archive`	SAE Identifiant service d'archive
`sae_originating_agency`	SAE Identifiant service producteur
`sae_numero_aggrement`	SAE Accord de versement

Les paramètres suivants ne sont disponibles que pour le super Admin ou un administrateur de groupe.

Nom paramètre	Description
`authority_group_id`	Identifiant du groupe dont fait partie la collectivité
`helios_ftp_login`	HELIOS ftp login
`helios_ftp_password`	HELIOS ftp mot de passe
`helios_ftp_dest`	HELIOS ftp Dest
`ext_siret`	Numéro de EXT SIRET
`module`	Tableau associatif de type clé => Valeur avec les clés, les identifiants de module retournés par la fonction admin_module.php et comme valeur : 0 : module désactivé , 1 : module activé

3.6. Liste des SIREN possibles

Nom du script : `/admin/authorities/admin_authorities_siren.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super admin ou administrateur de groupe

Description : permet de lister les SIREN autorisés pour créer une collectivité

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`authority_group_id`	Obligatoire pour le super admin. Interdit pour l'admin de groupe	entier	Identifiant du groupe

Retour : Une chaîne JSON contenant un tableau contenant la liste des SIREN autorisés.

3.7. Édition d'une collectivité

Nom du script : `/admin/authorities/admin_authority_edit_handler.php`

Méthode HTTP d'appel : POST

Droits nécessaires : Super admin, administrateur de groupe ou administrateur de collectivité

Description : permet de créer ou de modifier une collectivité

Paramètres à fournir :

Les lignes grisées indiquent les lignes qui ne peuvent être modifiées ou crées que par le super admin ou l'administrateur de groupe.

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`name`	obligatoire	chaine	Nom de la collectivité
`siren`	obligatoire	numéro SIREN	numéro SIREN autorisé pour ce groupe (voir fonction `admin_authorities_siren.php`)
`authority_groupe_id`	obligatoire	entier	Identifiant du groupe auquel est rattachée cette collectivité. Facultatif pour un admin de groupe.
`email`	facultatif	chaine	email de la collectivité
`default_broadcast_email`	facultatif	chaine	email de notification coché par défaut pour ACTES (séparé par des virgules)
`broadcast_email`	facultatif	chaine	Autres emails de notification (séparés par des virgules)
`status`	obligatoire	0 ou 1	0 : désactivé , 1 : activé
`authority_type_id`	obligatoire	entier	Identifiant du type de collectivité (voir la fonction `admin_authority_types.php`)
`address`	facultatif	chaine	Adresse de la collectivité
`postal_code`	facultatif	chaine	Code Postal
`city`	facultatif	chaine	Ville
`department`	obligatoire	chaine	Département sur trois chiffres (ex : 069 ; 972; 02B)
`district`	obligatoire	entier	numéro de l'arrondissement (sur un chiffre)
`telephone`	facultatif	chaine	Numéro de téléphone
`fax`	facultatif	chaine	Numéro de fax
`ext_siret`	facultatif	chaine	Numéro SIRET
`helios_ftp_login`	facultatif	chaine	Login FTP pour HELIOS
`helios_ftp_password`	facultatif	chaine	Mot de passe FTP pour HELIOS
`helios_ftp_dest`	facultatif	chaine	HELIOS ftp Dest
`email_mail_securise`	facultatif	chaine	Email source des mails sécurisés émis par la collectivité.
`perm_XXX`	obligatoire	0 ou 1	Indique si le module XXX est activé. Voir la fonction `admin_modules.php` .

> *Remarque* : Il est nécessaire de fournir à chaque fois l'ensemble des paramètres pour une modification, ceux-ci peuvent être obtenus grâce à la fonction `admin_authority_detail.php`

Retour :

Une chaîne JSON contenant un status :

- `error` : dans ce cas, msg-erreur contient le message de l'erreur.
- `ok` : dans ce cas, id contient l'identifiant de la collectivité créée ou modifiée.

3.8. Liste des rôles des utilisateurs

Nom du script : `/admin/users/admin_users_role.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super admin ou administrateur de groupe ou administrateur de collectivité

Description : permet de lister les rôles des utilisateurs

Paramètres à fournir : Aucun

Retour :

Une chaîne JSON contenant un tableau contenant des lignes de type id => description

Nom paramètre	Description
`id`	Identifiant interne du rôle
`description`	Description du rôle

3.9. Liste des utilisateurs

Nom du script : `/admin/users/admin_users.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super Admin ou administrateur de groupe ou administrateur de collectivité

Description : permet de lister les collectivités de la plateforme.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`name`	facultatif	chaîne	Permet de filtrer la liste des utilisateurs. Les noms des utilisateurs récupérés contiennent la chaîne « name »
`role`	facultatif	chaîne	Filtre suivant un des identifiants de rôle retourné par la fonction `admin_users_role.php`
`group`	facultatif	entier	Filtre les utilisateurs dont la collectivité appartient au groupe dont l'identifiant est donné par ce paramètre. Ce paramètre n'est utilisable que pour le super admin. Les identifiants de groupe peuvent être récupérés via la fonction « `admin_groups.php` »
`authority`	facultatif	entier	Filtre les utilisateurs appartenant à la collectivité dont l'identifiant est donné par ce paramètre. Ne fonctionne pas pour un admin de collectivité.

Retour :

une chaîne JSON contenant un tableau dont les lignes regroupent les informations sur les utilisateurs.

Nom paramètre	Description
`id`	Identifiant unique de l'utilisateur
`name`	Nom de l'utilisateur

`givenname`	Prénom
`email`	email de l'utilisateur
`role`	Rôle de l'utilisateur, voir la fonction `admin_users_role`
`status`	0 : désactivé , 1 : activé
`authority_id`	Identifiant de la collectivité à laquelle appartient l'utilisateur
`authority_name`	Nom de la collectivité

3.10. Détail d'un utilisateur

Nom du script : `/admin/users/admin_user_detail.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super Admin, Admin de groupe, ou Admin de collectivité

Description : permet d'obtenir l'ensemble des informations sur un utilisateur

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant de l'utilisateur

Retour :

une chaîne JSON contenant un tableau associatif contenant les informations sur l'utilisateur.

Nom paramètre	Description
`id`	Identifiant unique de l'utilisateur
`name`	Nom
`givenname`	Prénom
`login`	Login de l'utilisateur (si il y a plusieurs utilisateurs pour le même certificat)
`email`	email de l'utilisateur
`telephone`	Téléphone de l'utilisateur
`status`	0 : Désactivé , 1 : Activé
`authority_id`	Identifiant de la collectivité de l'utilisateur
`authority_group_id`	Identifiant du groupe dont fait partie la collectivité de l'utilisateur
`role`	Rôle de l'utilisateur (voir `admin_users_role.php`)
`certificate`	Le contenu du certificat de l'utilisateur au format PEM
`module`	Tableau associatif avec comme clé l'identifiant des modules (voir admin_modules.php) et comme valeur une des valeurs suivantes : - NONE : pas de droit , - RO : lecture seule , - RW : Modification , - GRANT : concession
`other_id`	Tableau contenant les autres identifiants d'utilisateur qui partagent ce certificat

3.11. Édition d'un utilisateur

Nom du script : `/admin/users/admin_users_edit_handler.php`

Méthode HTTP d'appel : POST

Droits nécessaires : Super admin ou administrateur de groupe ou administrateur de collectivité

Description : permet de créer ou de modifier les propriétés d'un utilisateur.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`id`		entier	Identifiant de l'utilisateur à modifier. Si aucun, il s'agit de la création d'un utilisateur
`name`	obligatoire	chaîne	Nom de l'utilisateur
`givenname`	obligatoire	chaîne	Prénom
`email`	facultatif	chaîne	Email
`telephone`	facultatif	chaîne	Téléphone
`status`	facultatif	0 ou 1	0 : désactivé , 1 : activé
`authority_id`	obligatoire	entier	Identifiant de la collectivité. Ce paramètre n'est pas modifiable une fois l'utilisateur créé. Il dépend également des droits de l'utilisateur qui utilise l'API (il ne sert à rien pour un admin de collectivité)
`role`	obligatoire	chaîne	Un des roles suivants : SADM (super admin), GADM (admin de groupe), ADM (admin de collectivité), USER (utilisateur). Il n'est bien sur pas possible de créer un utilisateur avec un rôle dont on ne dispose pas soi-même.
`authority_group_id`	obligatoire	entier	Identifiant du groupe. Ne sert que pour la création d'un utilisateur par le super admin. Ne peut pas être modifié.
`certificate`	obligatoire	fichier PEM	Fichier contenant le certificat de l'utilisateur. Obligatoire lors d'une création (sauf à utilisé `new_id`, voir plus loin). Facultatif lors d'une modification.
`login`	facultatif	chaîne	Login de l'utilisateur pour un partage de certificat. Obligatoire dans ce cas.
`password`	facultatif	chaîne	Mot de passe correspondant.
`new_id`	facultatif	entier	Permet de copier un certificat pour lui attribuer un nouvel utilisateur. new_id représente l'identifiant de l'utilisateur dont on veut copier le certificat. Cet utilisateur, comme celui qu'on va créer doivent impérativement disposer chacun d'un login/mot de passe
`perm_XXX`	facultatif	chaîne	XXX représente un numéro de module (voir `admin_modules.php`) et comme valeur une des valeurs suivantes : NONE (pas de droit), RO (lecture seule), RW (Modification), GRANT (concession). La possibilité de modifier les paramètres dépend de nos propres droits et des droits de la collectivité ou du groupe à utiliser tel ou tel module.

> *Remarque* : Il est nécessaire de fournir à chaque fois l'ensemble des paramètres pour une modification, ceux-ci peuvent être obtenus grâce à la fonction `admin_users_detail.php`

Retour :

Une chaîne JSON contenant un status :

- `error` : dans ce cas, error-message contient le message de l'erreur.
- `ok` : dans ce cas, id contient l'identifiant de l'utilisateur créé ou modifié.

3.12. Ajout d'un SIREN

Nom du script : `/admin/authorities/admin_authorities_add_siren.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super admin ou administrateur de groupe

Description : permet d'ajouter un siren dans un groupe

Paramètres à fournir :

Nom paramètre	Présence	Format	Description

`authority_group_id`	obligatoire	entier	Correspond à l'id du groupe dans lequel on souhaite ajouter le siren
`siren`	obligatoire	entier de 9 chiffres	SIREN a ajouter dans le groupe

Retour :

Une chaîne JSON contenant un status :

- `error` : dans ce cas, error-message contient le message de l'erreur.(Echec de l'authentification, Accès refusé, identifiant groupe invalide, siren non valide, siren déjà présent)
- `ok` : dans ce cas, message contient « ajout réussi »

3.13. Liste des numéros de SIRET

Nom du script : `/admin/authorities/admin_authority_siret.php`

Méthode HTTP d'appel : GET

Droits nécessaires : Super admin ou administrateur de groupe

Description : permet de lister les numéros de SIRET à une collectivité.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`siret`	obligatoire	entier de 14 chiffres	Numéro SIRET

Retour :

Une chaîne JSON contenant :

- `error` : dans ce cas, error-message contient le message de l'erreur.(Echec de l'authentification, Accès refusé, identifiant groupe invalide, siren non valide, siren déjà present), ou
- la liste des SIRET de la collectivité

3.14. Ajout d'un numéro SIRET

Nom du script : `/admin/authorities/admin_authority_siret_add.php`

Méthode HTTP d'appel : POST

Droits nécessaires : Super admin ou administrateur de groupe

Description : permet d'ajouter un numéro de SIRET à une collectivité.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`authority_id`	obligatoire	entier	Identifiant de la collectivité
`siret`	obligatoire	entier de 14 chiffres	Numéro SIRET

Retour :

Une chaîne JSON contenant un status :

- `error` : dans ce cas, error-message contient le message de l'erreur.(Echec de l'authentification, Accès refusé, identifiant groupe invalide, siren non valide, siren déjà présent)
- `ok` : dans ce cas, message contient « Numéro de SIRET ajouté »

3.15. Supression d'un numéro SIRET

Nom du script : `/admin/authorities/admin_authority_siret_del.php`

Méthode HTTP d'appel : POST

Droits nécessaires : Super admin ou administrateur de groupe

Description : permet de supprimer un numéro de SIRET à une collectivité.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`id`	obligatoire	entier	Identifiant de la collectivité
`siret`	obligatoire	entier de 14 chiffres	Numéro SIRET

Retour :

Une chaîne JSON contenant un status :

- `error` : dans ce cas, error-message contient le message de l'erreur.(Echec de l'authentification, Acces refusé, identifiant groupe invalide, siren non valide, siren déjà présent)
- `ok` : dans ce cas, message contient « Numéro de SIRET retiré»

3.16. Ajouter un service

Cette fonction est disponible à partir de la version 3.0.13 de S²LOW

Nom du script : `admin/services/add-service-user.php`

Méthode HTTP d'appel : POST

Droits nécessaires : administrateur (super, groupe ou collectivité)

Description : permet d'ajouter un service à une collectivité

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`authority_id`	obligatoire	entier	Identifiant de la collectivité
`name`	obligatoire	chaine de 128 caractères maximum	Nom du service à ajouter

Retour :

Une chaîne JSON contenant un status :

- `error` : dans ce cas, error-message contient le message de l'erreur.(Echec de l'authentification, Acces refusé, service déjà existant)
- `ok`

3.17. Lister les services

Cette fonction est disponible à partir de la version 3.0.13 de S²LOW

Nom du script : `admin/services/list-service.php`

Méthode HTTP d'appel : GET

Droits nécessaires : administrateur (super, groupe ou collectivité)

Description : Liste les services associés à une collectivité

Paramètres à fournir :

Nom paramètre	Présence	Format	Description

`authority_id`	obligatoire	entier	Identifiant de la collectivité

Retour :

Une chaîne JSON contenant un status :

- `error` : dans ce cas, error-message contient le message de l'erreur.(Echec de l'authentification, Acces refusé)
- ou bien un fichier json contenant les informations sur les services (id = l'identifiant du service)

3.18. Ajouter un utilisateur dans un service

Cette fonction est disponible à partir de la version 3.0.13 de S²LOW

Nom du script : `admin/users/add-user-to-service.php`

Méthode HTTP d'appel : POST

Droits nécessaires : administrateur (super, groupe ou collectivité)

Description : Ajoute un utilisateur dans un service

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier.
`id_service`	obligatoire	entier	Identifiant du service retourné par la fonction list-service.php
`id_user`	obligatoire	entier	Identifiant de l'utilisateur

Retour :

Une chaîne JSON contenant un status :

- `error` : dans ce cas, error-message contient le message de l'erreur.(Echec de l'authentification, Acces refusé, service déjà existant)
- `ok`

4. DESCRIPTION DE L'API HTTPS POUR ACTES

L'API HTTPS consiste en des scripts PHP qui sont appelés par des méthodes HTTP GET ou POST.

Tous les chemins de scripts indiqués par la suite sont relatifs à l'adresse de base du serveur (qui est de la forme https://nom.du.serveur/).

Certains scripts ont besoin de savoir que l'appel est effectué par une application métier afin d'adapter la réponse retournée dans un format plus facilement exploitable par une application (par défaut les scripts retournent des redirections HTTP vers une page qui affiche le résultat du traitement). C'est à cette fin que ces scripts acceptent dans les requêtes un paramètre nommé « api » qui prend pour valeur « 1 ». Si ce paramètre n'est pas présent ou initialisé à une autre valeur, le résultat retourné est formaté pour un client navigateur et non une application métier.

Dans la suite nous décrivons chaque service avec son script API, la méthode et les paramètres, ainsi que le retour du serveur.

4.1. Postage d'une transaction de transmission d'acte

Nom du script : `/modules/actes/actes_transac_create.php`

Méthode HTTP d'appel : `POST` avec `enctype="multipart/form-data"`

Description : Soumission d'un acte à transmettre au ministère. Nécessite la fourniture de toutes les informations concernant l'acte à transmettre. La plate-forme se charge de générer l'archive .tar.gz et de la poster au ministère.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier. Change le type de retour fourni par TéDéTIS.
`nature_code`	obligatoire	entier	Code nature de l'acte issu de la classification matière.
`classif1`	obligatoire	entier	Code classification niveau 1
`classif2`	obligatoire	entier	Code classification niveau 2
`classif3`	facultative	entier	Code classification niveau 3
`classif4`	facultative	entier	Code classification niveau 4
`classif5`	facultative	entier	Code classification niveau 5
`number`	obligatoire	15 caractères maxi, chiffres, lettres en majuscule ou _	Numéro de l'acte interne à la collectivité
`decision_date`	obligatoire	date ANSI : YYYY-MM-DD	Date de décision de l'acte
`subject`	obligatoire	texte longueur maxi 500 caractères	Objet de l'acte
`acte_pdf_file`	obligatoire	fichier attaché	Fichier PDF ou XML de l'acte
`acte_pdf_file_sign`	facultative	fichier attaché	Fichier de signature de l'acte au format PKCS7
`type_acte`	obligatoire(1)	5 caractères	Type de pièce jointe, un des codes qui apparait dans les attributs CodeTypePJ de la classification
`acte_attachments[]`	facultative	fichiers attachés	Fichiers de pièces jointes (PDF, JPG ou PNG). Ce paramètre peut apparaître plusieurs fois.
`acte_attachments_sign[]`	facultative	fichiers attachés	Fichiers de signature des pièces jointes. Ce paramètre peut apparaître plusieurs fois.
`type_pj[]`	obligatoire(1)	5 caractères	Type de pièce jointe, un des codes qui apparait dans les attributs CodeTypePJ de la classification
`en_attente`	facultative	entier de valeur 0 ou 1	0 (valeur par défaut) : la transaction sera envoyée en préfecture , 1 : la transaction est placée dans un état d'attente. Un opérateur devra la valider pour envoi à la préfecture.
			Permet à la collectivité d'indiquer si la télétransmission est multi-canal,

`document_papier`	facultative	entier de valeur 0 (défaut) ou 1	c'est-à-dire avec des documents complémentaires envoyés au format papier (1 dans ce cas)

> *Remarque* : Les deux variables `acte_attachments[]` et `acte_attachments_sign[]` sont vues comme des tableaux par la plate-forme et doivent être synchronisées. Le premier élément du tableau `acte_attachments_sign[]` doit être le fichier contenant la signature du premier élément du tableau `acte_attachments[]` et ainsi de suite.

> 1. *Remarque* : A compter du 08/06/2019, la typologie de la pièce principale et des annexes est obligatoire

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n`, numéro de la transaction créée sur la deuxième ligne
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.2. Postage d'une archive .tar.gz

Nom du script : `/modules/actes/actes_transac_submit.php`

Méthode HTTP d'appel : `POST` avec `enctype="multipart/form-data"`

Description : Soumission d'une archive .tar.gz contenant une ou plusieurs transactions à transmettre au ministère. Nécessite la fourniture d'un fichier .tar.gz. La plate-forme se charge de valider l'archive et de la poster au ministère.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier. Change le type de retour fourni par TéDéTIS.
`enveloppe`	obligatoire	fichier attaché	Fichier archive .tar.gz contenant les transactions

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n`, nombre de transactions créées sur la deuxième ligne terminée par `\n`, numéros des transactions crées sur les lignes suivantes (dans le même ordre que dans l'enveloppe XML de l'archive)
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.3. Transaction de demande d'annulation d'acte

Nom du script : `/modules/actes/actes_transac_cancel.php`

Méthode HTTP d'appel : `POST`

Description : Génération d'une transaction de demande d'annulation d'un acte.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier. Change le type de retour fourni par TéDéTIS.
`id`	obligatoire	entier	Numéro d'identifiant de la transaction à annuler (numéro retourné lors de la création d'un acte)

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n`, numéro de la transaction d'annulation créée sur la deuxième ligne
- en cas d'échec : KO sur la première ligne terminée par `\n` message d'erreur sur les lignes suivantes.

4.4. Clôturer une transaction

Nom du script : `/modules/actes/actes_transac_close.php`

Méthode HTTP d'appel : `POST`

Description : Passer une transaction de transmission d'acte dans l'état « Validé » ou « Annulé ».

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier. Change le type de retour fourni par TéDéTIS.
`id`	obligatoire	entier	Numéro d'identifiant de la transaction à clôturer (numéro retourné lors de la création d'un acte)
`status`	obligatoire	chaîne	Statut dans lequel passer la transaction : `valid` : passer en état « Validé » ; `invalid` : passer en état « Refusé ».

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n`.
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.5. Transaction de demande de classification

Nom du script : `/modules/actes/actes_classification_request.php`

Méthode HTTP d'appel : `POST`

Description : Génération d'une transaction de demande de classification matières/sous-matières.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier. Change le type de retour fourni par TéDéTIS.

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n`, numéro de la transaction de demande de classification créée sur la deuxième ligne.
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.6. Récupérer le fichier XML de la classification matières/sous-matières

Nom du script : `/modules/actes/actes_classification_fetch.php`

Méthode HTTP d'appel : `GET`

Description : Récupérer le fichier XML de la dernière classification matières/sous-matières pour la collectivité.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier. Change le type de retour fourni par TéDéTIS.

Retour de l'appel :

- en cas de succès de l'appel : retour du fichier XML en attachement (nom de fichier : classification.xml)
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.7. Obtenir le statut d'une transaction

Nom du script : `/modules/actes/actes_transac_get_status.php`

Méthode HTTP d'appel : `GET`

Description : Obtention du code de statut d'une transaction.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`transaction`	obligatoire	entier	Identifiant de la transaction concernée.

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n`, statut de la transaction sur la deuxième ligne. Liste des statuts :
  - -1 : Erreur,
  - 0 : Annulé,
  - 1 : Posté,
  - 2 : En attente de transmission,
  - 3 : Transmis,
  - 4 : Acquittement reçu,
  - 5 : Validé,
  - 6 : Refusé ;

  Pour le statut 4, les lignes suivantes contiennent le contenu du fichier d'acquittement (ARActe). Pour le statut -1, la troisième ligne contient le message d'erreur.

- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.8. Obtenir un numéro de série pour génération enveloppe

Nom du script : `/modules/actes/actes_transac_get_env_serial.php`

Méthode HTTP d'appel : `GET`

Description : Obtention d'un numéro de série unique pour génération d'une enveloppe.

Paramètres à fournir :

- Aucun

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n`, numéro de série unique sur la deuxième ligne
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.9. Définir l'URL d'archivage d'une transaction

Nom du script : `/modules/actes/actes_transac_set_archive_url.php`

Méthode HTTP d'appel : `POST`

Description : Définition de l'URL d'archivage d'une transaction.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant de la transaction concernée.
`url`	obligatoire	chaîne	URL d'archivage de la transaction
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier. Change le type de retour fourni par TéDéTIS.

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n`.
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.10. Récupérer la liste des documents du ministère relatifs à un ACTE

Nom du script : `/modules/actes/actes_transac_get_document.php`

Méthode HTTP d'appel : `GET`

Description : Définition de l'URL d'archivage d'une transaction.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant de la transaction concernée.

Retour de l'appel :

- en cas de succès de l'appel : Liste de l'ensemble des documents relatifs à l'acte OU BIEN le document lui-même si la transaction est une transaction de document relatif. La liste contient une ligne par document avec trois numéros représentant :
- le type de document (2, 3, 4 ou 5)
    - 2 : Courrier simple
    - 3 : Demande de pièces complémentaires
    - 4 : Lettre d'observation
    - 5 : Déféré au tribunal administratif.
- le statut du document (7 ou 8 pour les documents reçus, 1 à 5 pour les documents envoyés) Les numéros 1 à 5 ont la même signification que pour la transmission d'un acte original.
- l'identifiant interne de la transaction (numérique)
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.11. Poster une réponse à un document du ministère relatif à un ACTE

Nom du script : `/modules/actes/actes_transac_reponse_create.php`

Méthode HTTP d'appel : `POST`

Description : Définition de l'URL d'archivage d'une transaction.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant de la transaction concernée.
`api`	obligatoire	entier de valeur 1	Indique à la plate-forme que l'appel se fait par une application métier. Change le type de retour fourni par TéDéTIS.
`type_envoie`	Obligatoire pour les messages de type « Pièce complémentaire » ou « Lettre d'observation »	Entier (3 ou 4)	3 : Refus d'envoi de pièce , 4 : envoi de pièce
`acte_pdf_file`	obligatoire	fichiers attachés	Fichier PDF de la réponse
`acte_attachments[]`	Facultative uniquement pour les « Pièces complémentaires »	Fichiers attachés	Fichiers de pièces jointes (PDF, JPG ou PNG). Ce paramètre peut apparaître plusieurs fois.

Retour de l'appel :

- en cas de succès de l'appel : OK sur la première ligne terminée par `\n` les documents relatifs à l'acte sur les lignes suivantes.
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.12. Récupérer l'acte tamponné

Cette fonction est dépréciée dans la version 2.0 de S²LOW. Il n'est plus certain qu'elle soit disponible dans les versions ultérieures de S²LOW.

Nom du script : `/modules/actes/actes_transac_get_tampon.php`

Méthode HTTP d'appel : `GET`

Description : Récupération du fichier acte, tamponné par S²low

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`transaction`	obligatoire	entier	Identifiant de la transaction concernée.

Retour de l'appel :

- en cas de succès de l'appel : l'acte tamponné
- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.13. Récupérer la liste des documents associés à un acte

Nom du script : `/modules/actes/actes_transac_get_files_list.php`

Méthode HTTP d'appel : `GET`

Description : Récupération de la liste des fichiers associés à un acte (fichier métier, l'acte lui-même et les annexes)

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`transaction`	obligatoire	entier	Identifiant de la transaction concernée.

Retour de l'appel :

- en cas de succès de l'appel : un fichier JSON contenant un tableau des documents disponibles.
  Chaque document contient :
  - un id (permettant de récupérer le document, cf, fonction suivante)
  - name : le nom original du fichier
  - posted_filename : le nom du fichier, tel que posté par l'utilisateur (si posté par l'utilisateur).
  - mimetype : le type de contenu
  - size : la taille en octet
  - signature : la signature détachée au format PKCS#7 si disponible

On notera que le premier fichier est toujours le fichier métier XML, le second est l'acte lui-même et les suivants, les annexes attachées à l'acte.

- en cas d'échec : KO sur la première ligne terminée par `\n`, message d'erreur sur les lignes suivantes.

4.14. Récupérer l'acte (ou ses annexes), tamponné ou non

Nom du script : `/modules/actes/actes_download_file.php`

Méthode HTTP d'appel : `GET`

Description : Récupération un fichier attaché à un acte.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`file`	obligatoire	entier	Identifiant du fichier tel que retourné par la fonction `actes_transac_get_files_list.php`
`tampon`	non (false par défaut)	Boolean	Indique si le fichier doit être tamponné (ne fonctionne que pour les fichiers au format PDF)

`date_affichage`	non	Date au format Y-m-d	Ajoute la date d'affichage de l'acte dans le tampon

Retour de l'appel :

- le contenu du fichier lui-même.
- une erreur 404 peut-être retournée en cas d'échec.

4.15. Ordonner la télétransmission de l'acte [Appelé par le client]

Nom du script : `/modules/actes/actes_transac_post_confirm_api.php`

Méthode HTTP d'appel : `GET`

Description : Ordonne la télétransmission pour un acte qui est dans l'état « en attente d'être posté. » Cette fonction permet à un site tiers de préparer la télétransmission, puis de la faire télétransmettre en direct par un utilisateur sur S2low. On peut par exemple utiliser cela pour permettre à l'utilisateur de présenter un certificat RGS** pour assurer la télétransmission.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant de la transaction.
`url_return`	obligatoire	URL	Lors de la terminaison de cette fonction, on redirige l'utilisateur vers cette url. L'url peut contenir : %%ERROR %% qui est remplacé par 0 en cas de réussite et 1 en cas d'erreur. %%MESSAGE %% qui contiendra le message d'erreur.

Retour de l'appel : N/A

4.16. Ordonner la télétransmission de plusieurs actes [Appelé par le client]

Nom du script : `/modules/actes/actes_transac_post_confirm_api.php`

Méthode HTTP d'appel : `GET`

Description : Ordonne la télétransmission pour un acte qui est dans l'état « en attente d'être posté. » Cette fonction permet à un site tiers de préparer la télétransmission, puis de la faire télétransmettre en direct par un utilisateur sur S2low. On peut par exemple utiliser cela pour permettre à l'utilisateur de présenter un certificat RGS** pour assurer la télétransmission.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`id[]`	obligatoire	entier	Tableau d'identifiant des transactions.
`url_return`	obligatoire	URL	Lors de la terminaison de cette fonction, on redirige l'utilisateur vers cette url. Aucune erreur n'est générée ni retournée par la fonction. C'est au système appelant de vérifier que ces actes ont été télétransmis ou non.

Retour de l'appel : N/A

4.17. Récupérer la liste des statuts actes possibles

Nom du script : `/modules/actes/api/actes_status.php`

Méthode HTTP d'appel : `GET`

Description : Récupère les statuts des actes disponibles sur la plateforme

Paramètres à fournir : aucun

Retour de l'appel : un fichier JSON contenant la liste des status (identifiant du statut et libellé)

Exemple de retour:

```
{
  "-1": "Erreur",
  "0": "Annul\u00e9",
  "1": "Post\u00e9",
  "2": "En attente de transmission",
  "3": "Transmis",
  "4": "Acquittement re\u00e7u",
  "5": "Valid\u00e9",
  "6": "Refus\u00e9",
  "7": "Document re\u00e7u",
  "8": "Acquittement envoy\u00e9",
  "9": "Document envoy\u00e9",
  "10": "Refus d'envoie",
  "11": "Acquittement de document re\u00e7u",
  "12": "Envoy\u00e9 au SAE",
  "13": "Archiv\u00e9 par le SAE",
  "14": "Erreur lors de l'archivage",
  "15": "Re\u00e7u par le SAE",
  "16": "D\u00e9truite",
  "17": "En attente d'\u00eatre post\u00e9",
  "18": "En attente d'\u00eatre sign\u00e9",
  "19": "En attente de transmission au SAE",
  "20": "Erreur lors de l'envoi au SAE",
  "21": "Document re\u00e7u (pas d'AR)"
}
```


4.18. Récupérer le nombre d'actes dans un certain statut

Nom du script : `/modules/actes/api/number_actes.php`

Méthode HTTP d'appel : `GET`

Description : Récupère le nombre d'actes dans un certain status en fonction de sa collectivité de rattachement

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`status_id`	facultatif (0 par défaut)	entier	Identifiant du statut (cf actes_status.php).

Retour de l'appel : un fichier JSON contenant les informations de la requête et le nombre de transactions dans le statut sélectionné.

Exemple de retour:


```
{
  "status_id": 4,
  "authority_id": 2,
  "nb_transactions": 80
}
```


4.19. Récupérer les actes dans un certain statut

Nom du script : `/modules/actes/api/list_actes.php`

Méthode HTTP d'appel : `GET`

Description : Récupère les actes dans un certain status en fonction de sa collectivité de rattachement

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`status_id`	facultatif (0 par défaut)	entier	Identifiant du statut (cf actes_status.php).
`offset`	facultatif (0 par défaut)	entier	Identifiant du premier élément à récupérer
`limit`	facultatif (100 par défaut)	entier	Nombre de transaction à récupérer

Retour de l'appel : un fichier JSON contenant les informations sur les transactions actes dans le statut sélectionné.

Exemple de retour:


```
{
  "status_id": "4",
  "authority_id": "2",
  "offset": "0",
  "limit": "2",
  "transactions": [
```

```
    {
      "id": "158",
      "subject": "45456",
      "number": "201709111149",
      "date": "2017-09-01",
      "nature_descr": "Actes individuels",
      "classification": "3.5",
      "type": "1"
    },
    {
      "id": "157",
      "subject": "5465",
      "number": "201709111135",
      "date": "2017-09-01",
      "nature_descr": "Actes r\u00e9glementaires",
      "classification": "2.3",
      "type": "1"
    }
  ]
}
```


4.20. Récupérer la liste de documents de la préfecture

Cette fonction est disponible à partir de la version 3.0.15

Nom du script : `/modules/actes/api/list_document_prefecture.php`

Méthode HTTP d'appel : `GET`

Description : Permet de lister l'ensemble des documents non-lu envoyés par la préfecture (message 2-1, 3-1, 4-1 et 5-1). Il faut marquer les transactions récupérer comme lu afin de ne pas les récupérer à nouveau avec la fonction **marquer un document comme lu**

Paramètres à fournir : aucun

Retour de l'appel : un fichier JSON contenant la liste des transactions

Nom de la clé	Description
`id`	Identifiant de la transaction.
`type`	type du message selon la norme actes (exemple : 3 = demande de pièces complémentaires
`related_transaction_id`	Identifiant de la transaction original (message 1-1) de s2low concerné par le message
`number`	Numéro interne de l'acte (message 1-1)
`unique_id`	Numéro unique de l'acte (message 1-1)
`last_status_id`	Status de la transaction envoyé par la préfecture

Exemple de retour:


```
[
  {
    "id": "56",
    "type": "3",
    "related_transaction_id": "55",
    "number": "201805281136",
    "unique_id": "002-000000000-20180501-201805281136-AR",
    "last_status_id": "8"
  },
  {
    "id": "103",
    "type": "3",
    "related_transaction_id": "102",
    "number": "201808201712",
    "unique_id": "02A-000000000-20180801-201808201712-AI",
    "last_status_id": "8"
  },
  {
    "id": "106",
    "type": "2",
    "related_transaction_id": "105",
    "number": "201808211106",
    "unique_id": "02A-000000000-20180801-201808211106-AI",
    "last_status_id": "21"
  },
  {
    "id": "107",
    "type": "3",
    "related_transaction_id": "105",
    "number": "201808211106",
    "unique_id": "02A-000000000-20180801-201808211106-AI",
```

```
    "last_status_id": "8"
  },
  {
    "id": "108",
    "type": "5",
    "related_transaction_id": "105",
    "number": "201808211106",
    "unique_id": "02A-000000000-20180801-201808211106-AI",
    "last_status_id": "21"
  },
  {
    "id": "109",
    "type": "4",
    "related_transaction_id": "105",
    "number": "201808211106",
    "unique_id": "02A-000000000-20180801-201808211106-AI",
    "last_status_id": "8"
  }
]
```


4.21. Marquer un document comme lu

Cette fonction est disponible à partir de la version 3.0.15

Nom du script : `/modules/actes/api/document_prefecture_mark_as_read.php`

Méthode HTTP d'appel : `GET`

Description : Permet de marquer un document de la préfecture comme lu pour ne pas le récupérer à nouveau via la fonction **récupérer la liste de documents de la préfecture**

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`transaction_id`	obligatoire	entier	Identifiant de la transaction.

Retour de l'appel : un fichier JSON contenant le resultat (OK)

Exemple de retour:


```
{"result":"ok"}
```

5. DESCRIPTION DE L'API POUR LE MODULE HÉLIOS

5.1. Postage d'un fichier XML

Nom du script : `/modules/helios/api/helios_importer_fichier.php`

Méthode HTTP d'appel : `POST` avec enctype="multipart/formdata"

Description : Soumission d'un fichier XML représentant le document à transmettre à un comptable. Nécessite la fourniture d'un fichier .xml. La plateforme se charge de valider le fichier et de le poster au destinataire.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`enveloppe`	obligatoire	fichier attaché	Fichier XML qui représente le document financier

Retour de l'appel (fichier XML):

```
<import>
<id> numéro de la transaction créée </id>
<resultat> OK ou KO </resultat>
<message> message complémentaire </message>
</import>
```

Attention : une signature invalide donnera lieu à un OK pour notifier la création de la transaction mais cette dernière apparaitra immédiatement en statut erreur (1).

5.2. Récupération du statut d'une transaction

Nom du script : `/modules/helios/api/helios_transac_get_status.php`

Méthode HTTP d'appel : `GET`

Description : Obtention du code de statut d'une transaction.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`transaction`	obligatoire	entier	Identifiant de la transaction concernée

Retour de l'appel : fichier xml

```
<transaction>
<id> numéro de la transaction </id>
<resultat> OK ou KO </resultat>
<status> statut </status>
<message> message complémentaire </message>
</transaction>
```

Liste des statuts :

Code	Description
-1	Erreur
1	Posté
2	En attente de transmission
3	Transmis
4	Ack
6	Nack
7	En traitement
8	Information disponible
9	Envoye au SAE
10	Accepter par le SAE

11	Refuser par le SAE
13	En attente d'être signée
14	En attente d'être postée
19	En attente de transmission au SAE
20	Erreur lors de l'envoi au SAE

5.3. Récupération du PES ACQUIT

Nom du script : `/modules/helios/api/helios_download_acquit.php`

Méthode HTTP d'appel : `GET`

Description : Récupération de l'acquitement *PES ACQUIT* correspondant à un *PES ALLER*

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant de la transaction concernée.

Retour de l'appel : Directement le fichier PES ACQUIT

5.4. Récupération de la liste des PES_RETOUR

Nom du script : `/modules/helios/api/helios_get_list.php`

Méthode HTTP d'appel : `GET`

Description : Obtention de la liste des PES_RETOUR avec l'état « non lu »

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`collectivite`	obligatoire	entier	Identifiant de la collectivité concernée.

Retour de l'appel : fichier xml

```
<liste>
<idColl> identifiant de la collectivité </idColl>
4/5
Documentation API HTTPS – TéDéTIS Ed1.1.Q
<dateDemande> date de demande de la liste </dateDemande>
<resultat> OK ou KO </resultat>
<message> message complémentaire </message>
<pes_retour>
<id> identifiant du pes_retour </id>
<nom> nom </nom>
<date> date de réception du pes_retour par le tdt </date>
</pes_retour>
<pes_retour>
<id> identifiant du pes_retour </id>
<nom> nom </nom>
<date> date de réception du pes_retour par le tdt </date>
</pes_retour>
...
</liste>
```

5.5. Changement d'état d'un PES_RETOUR

Nom du script : `/modules/helios/api/helios_change_status.php`

Méthode HTTP d'appel : `GET`

Description : Changement d'état de PES_RETOUR : passage de l'état non lu à l'état lu.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`idRetour`	obligatoire	entier	Identifiant des pes_retour concernés

Retour de l'appel : fichier xml

```
<change>
<id> numéro de la transaction </id>
<resultat> OK ou KO </resultat>
<message> message complémentaire </message>
</change>
```

5.6. Récupération du contenu d'un PES_RETOUR

Nom du script : `/modules/helios/api/helios_get_retour.php`

Méthode HTTP d'appel : `GET`

Description : Récupération du fichier xml correspondant au pes_retour demandé.

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`idRetour`	obligatoire	entier	Identifiant des pes_retour concernés

Retour de l'appel : le fichier xml du pes_retour

6. DESCRIPTION DE L'API POUR LE MAIL SÉCURISÉ

Cette partie décrit les fonctions de l'API HTTP permettant d'accéder via un programme au module de mail sécurisé du logiciel S²low.

Tous les chemins de scripts indiqués dans cette partie sont relatifs à l'adresse de base du serveur (qui est de la forme https://nom.du.serveur/).

L'ensemble des scripts est situé dans le répertoire /modules/mail/api/

6.1. Version de l'API (mail sécurisé)

Nom du script : `/modules/mail/api/version.php`

Méthode HTTP d'appel : `GET`

Description : Renvoie la version de l'API mail sécurisé

Paramètres à fournir : aucun

Retour : Un fichier texte ligne à ligne (exemple ci-dessous).

```
version_api
detail_version_api
```

6.2. Liste des utilisateurs de la collectivité

Nom du script : `/modules/mail/api/user-mail.php`

Méthode HTTP d'appel : `GET`

Paramètres à fournir : aucun

Retour : Un fichier texte ligne à ligne ou chaque ligne désigne une entrée de l'annuaire. Les lignes sont formatées de la manière suivante :

```
id_utilisateur:email:description[:groupe[:groupe …]]
```

6.3. Nombre de mails sur le système

Nom du script : `/modules/mail/api/nb-mail.php`

Méthode HTTP d'appel : `GET``

Paramètres à fournir : aucun

Retour : Le nombre de mails (entier)

6.4. Liste de mails

Nom du script : `/modules/mail/api/list-mail.php`

Méthode HTTP d'appel : `GET`

Paramètres à fournir :

Nom paramètre	Présence	Format	Description
`limit`	facultative	entier	Nombre maximum de mails listés
`offset`	facultative	entier	Sauter offset mail avant l'affichage

Retour :

Le script affiche **limit** mails classés par date inverse de création à partir de l'email d'ordre**offset**.

Chaque ligne représente les informations sur un email de la manière suivante :

```
id_mail:date creation:état confirmation:objet
```

`état confirmation` est soit :

- aucune confirmation,
- confirmé partiellement,
- confirmé

6.5. Détail d'un email

Nom du script : `/modules/mail/api/detail-mail.php

Méthode HTTP d'appel : `GET`

Paramètre du script :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant du mail retourné par le script list-mail.php

Retour : un fichier contenant l'ensemble des informations sur le mail

```
id : *****
date_envoi : *****
password : *****
fn_download : *****
status : *****
objet : *****
emis: *****

==message==
*****
```

Nom paramètre	Description
`id`	identifiant du mail
`date_envoi`	date d'envoi du mail
`password`	mot de passe éventuel pour protéger le mail
`fn_download`	code md5 permettant de récupérer les fichiers
`status`	statut de confirmation (identique liste mail)
`objet`	sujet du mail
`emis`	"description" :champs input:confirmé:date_confirmation
`message`	le message lui-même

`champs input` est soit :

- mailTo
- mailCC
- mailBCC

`confirmé` est soit :

- t (vrai)
- f (faux)

`date_confirmation` est présent si `confirmé` est vrai

Pour récupérer les fichiers il suffit d'interroger :

`modules/mail/template/download.php?filename=mail.zip&root=fn_download`

6.6. Envoi d'un mail

Nom du script : `/modules/mail/api/send-mail.php

Méthode HTTP d'appel : `POST`

Paramètre du script :

Nom paramètre	Présence	Format	Description
`mailTo`	obligatoire	Liste d'adresse email	Destinataires
`mailCC`	facultatif	Liste d'adresse email	Mail en copie
`mailBCC`	facultatif	Liste d'adresse email	Mail en copie caché
`objet`	obligatoire	Chaîne	sujet du mail
`message`	obligatoire	Chaîne	message
`password`	facultatif	Chaîne	Mot de passe de protection du message
`uploadFileN`	facultatif	File	Fichier à attaché (autant que souhaité) uploadFile1 pour le premier, uploadFileN...

La liste des adresses email est formatée comme suit :

- chaque adresse est séparée par une virgule
- une adresse est soit une adresse unique, soit un groupe
- une adresse unique est écrite de la manière suivante :*"description" [email]*
- un groupe est écrit de la manière suivante :*nom du groupe (groupe)*

Exemple :

"eric" [eric@sigmalis.com], "pierre" [pierre@adullact.org], service comptable (groupe)

Retour :

- Si le mail a été accepté par le système :

```
ok:id_mail
```

- Sinon : `erreur:raison de l'erreur`

6.7. Suppression d'un email

Nom du script : `/modules/mail/api/delete-mail.php

Méthode HTTP d'appel : `GET`

Paramètre du script :

Nom paramètre	Présence	Format	Description
`id`	obligatoire	entier	Identifiant du mail retourné par le script list-mail.php

Retour :

- OK si le mail a été supprimé
- ERREUR suivie de la raison de l'erreur sinon.