Documentation technique de l’API KageSecur

Lien utile : faire un hash en SHA256

Connexion :

Chemin d’accès

L’application Web est versionnée de ce fait les chemins d’accès à celle-ci sont susceptibles d’évoluer. Pour la première version, l’accès est :

https://[nom du node]:42000/connexionKS

Forme de la requête

Le fonctionnement de l'API est le suivant : il suffit d’interroger cette dernière en passant des paramètres POST en format JSON dans le body de la requête : « empreinte » & « hashJwt ».

Il faut aussi ajouter un header “authorization” avec la donnée “noJwt”. Ce header doit être nommé de la façon suivante : “Bearer [jwt]”.

Exemple : “{headers: {authorization: "Bearer " + “noJwt”*}}”*

Explication des champs demandés :

empreinte = SHA256(identifiant+clef)

hashJwt = empreinte

exemple:


```
{
"empreinte":"467baa6c1a9337043bbf7837b4ab15022f8d5002c10947a844112ae988a5e910",
"hashJwt":"467baa6c1a9337043bbf7837b4ab15022f8d5002c10947a844112ae988a5e910"
}
```

Réponses de la requête

Les réponses suivantes peuvent être renvoyés par l'API :

La réponse à la requête est renvoyée sous format JSON. Si le code réponse est 200, voici la liste des données du flux de réponse :

- resultat : "Connecté" ou "Inconnu" ou "Abonnement expiré"
- gestionnaire : true/false
- jwt : code jwt utile pour effectuer les requêtes suivantes

-200 : requête OK

exemple de requête :


```
{

    "resultat": "Connecté",

    "gestionnaire": true,

    "jwt":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJoYXNoIjoiNmNhYmMyOWQ3NTM0MWJkMDUxODU4N2Y5N
TExNGYyYjljNzI3YmViMTYwZWRjNTk3ZmE2M2QxMTI0NjQwODk1ZSIsImV4cCI6MTYzNTMyNDg3Nzc2MCwi
aWF0IjoxNjM1MzIxMjc3NzYwfQ.ioZIZYsDhnwfDRxy8GWHWmBg9uvLzeJlQf6ad8klDP8"

}
```


-403 : requête interdite (problème d'authentification)

Enregistrement d’une preuve :

Chemin d’accès

L’application Web est versionnée de ce fait les chemins d’accès à celle-ci sont susceptibles d’évoluer. Pour la première version, l’accès est :

https://[nom du node]:42000/writeProof

Forme de la requête

Le fonctionnement de l'API est le suivant : il suffit d’interroger cette dernière en passant des paramètres POST dans le body de la requête : « empreinte », « hashJwt », « allDatas », “checkedExterne”, ”checkedVisible”

Il faut aussi ajouter un header “authorization” qui contient le jwt renvoyé lors de la connexion. Ce header doit être nommé de la façon suivante : “Bearer [jwt]”.

Exemple : “{headers: {authorization: "Bearer " + [jwt]}}”

Explication des champs demandés :

empreinte = SHA256(identifiant + clef)

hashJwt = empreinte

allDatas (un tableau d’objets) =

```
[{"empreinte":[empreinte du client], "hash":[hash du fichier], "fileName":[nom du fichier], "node":[url sur lequel on enregistre le document]},...]
```

exemple :

```
{

    "empreinte":"467baa6c1a9337043bbf7837b4ab15022f8d5002c10947a844112ae988a5e910",

     "hashJwt":"467baa6c1a9337043bbf7837b4ab15022f8d5002c10947a844112ae988a5e910",


"allDatas":[{"empreinte":"467baa6c1a9337043bbf7837b4ab15022f8d5002c10947a844112ae988a5e910",
"hash":"1055333b299feee2c67e46ccda15d0c96370dc6d0efe06b829b3ef9408683715",
"fileName":"image.png", "node":"localhost"}]

}
```

Réponses de la requête

La réponse à la requête est envoyée sous format JSON. Si le code réponse est 200, voici la liste des données du flux de réponse :

- resultat : “Utilisateur introuvable” ou un object contenant le résultat pour chaque document (= “Ce document est déjà enregistré.” ou true)
- jwt : code jwt utile pour effectuer les requêtes suivantes. Un jwt est renvoyé uniquement si celui utilisé est sous le point d’expirer.

-200 : requête OK


```
{

    "resultat": {

        "image.png": true

    }

}
```


-403 : requête interdite (problème d’authentification)

Vérification d’un document :

Chemin d’accès

L’application Web est versionnée de ce fait les chemins d’accès à celle-ci sont susceptibles d’évoluer. Pour la première version, l’accès est :

https://[nom du node]:42000/verification

Forme de la requête

Le fonctionnement del'API est le suivant : il suffit d’interroger cette dernière en passant des paramètres POST « empreinte », « hashJwt », « hash ».

Il faut aussi ajouter un header “authorization” qui contient le jwt renvoyé lors de la connexion. Ce header doit être nommé de la façon suivante : “Bearer [jwt]”.

Exemple : “{headers: {authorization: "Bearer " + [jwt]}}”

Explication des champs demandés :

empreinte = SHA256(identifiant + clef)

hashJwt = empreinte

hash = SHA256(sha256 du fichier)

Réponses de la requête

La réponse à la requête est envoyée sous format JSON. Si le code réponse est 200, voici la liste des données du flux de réponse :

- resultat : “Votre document est enregistré.” ou “Les informations ne sont pas conformes et ne peuvent donc pas être validées.” ou "Ce document ne vous appartient pas." ou "Le document n'a pas encore été validé." ou "Le document a été refusé par son validateur." ou "Ce document n'est pas enregistré."
- jwt : code jwt utile pour effectuer les requêtes suivantes. Un jwt est renvoyé uniquement si celui utilisé est sous le point d’expirer.

Si le résultat est “Votre document est enregistré.”, d’autres informations seront disponibles dans la réponse :

- date = date à laquelle le document a été enregistré
- validateur = nom du validateur ayant validé le document (s’il y en a)
- identifiant = pseudo du client possédant le document
- fileName = nom du fichier
- auth = sert à télécharger un certificat dans le site KageSecur

- hashFile = hash du fichier

-200 : requête OK


```
{
    "resultat": "Votre document est enregistré.",
    "date": "2021-10-27T08:12:51.783Z",
    "validateur": [],
    "identifiant": "johnDoe",
    "fileName": "image.png",
    "auth": "1635322596973162311",
    "hashfile": "8c470831917e833c54370bbbb1ce4434f9a2930a2886cb40d9ed3a4c7f4b17f4"
}
```


-403 : requête interdite (problème d'authentification)