Direction Clients et Territoires

Direction des Systèmes d'Information

Procédure de déchiffrement des documents émis par Enedis sur les canaux numériques d'un client entreprise

Identification : **Enedis-NOI-CF_107E**

Version : V3

Nb. de pages : 9

Résumé / Avertissement

Le chiffrement de données transportées via Internet permet de garantir la confidentialité des informations émises par Enedis vers un client entreprise. Ces données sont émises sur les canaux de contacts définis dans la publication (adresse courriel et/ou serveur FTP). Ce document décrit :

- les modalités de chiffrement par Enedis des publications chiffrées vers les canaux numériques d'un client entreprise
- les modalités d'obtention des clés utilisées lors de ces opérations de chiffrement/déchiffrement,
- les modalités de déchiffrement, par le client entreprise, des fichiers chiffrés reçus.

Document(s) associé(s) et annexe(s) :

Version	Date d'application	Nature de la modification	Annule et remplace
V1	01/06/2019	Création du document	N/A
V2	13/06/2023	Mise à jour du document et ajout d'un outil de déchiffrement	V1
V3	11/09/2024	Simplification et actualisation du document	V2

Accessibilité

☑ Libre ☐ Interne ☐ Restreinte ☐ Confidentielle

■ **Destinataires :** clients entreprise

SOMMAIRE

Procédure de déchiffrement des documents émis par Enedis sur les canaux numériques d'un client entreprise

1 — Généralités

Enedis émet des flux chiffrés pour publier des fichiers vers les clients, Collectivités et Acteurs de Marché. Les algorithmes de chiffrement utilisés par Enedis sont ceux reconnus en France comme assurant un haut niveau de confidentialité. Pour le chiffrement vers les clients entreprise, Enedis utilise l'algorithme AES (« *Advanced Encryption Standard* »), avec des clés de 256 bits.

Il n'est pas possible de lire un fichier reçu par mail ou sur un serveur FTP sans procéder au déchiffrement décrit dans cette procédure.

NB : les étapes décrites ci-dessous correspondent à un déchiffrement sur un environnement Windows.

2 — Déchiffrer un fichier de données d'Enedis

Ce chapitre décrit un mode opératoire pour déchiffrer les fichiers de données reçus d'Enedis. Pour tout complément technique, vous pouvez vous référer aux annexes de ce document.

Les étapes nécessaires au déchiffrement sont les suivantes :

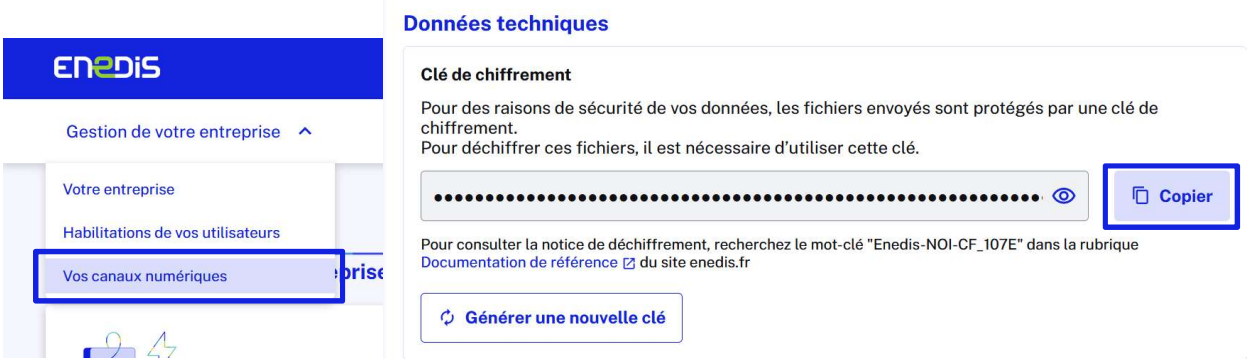
1. Avoir un fichier chiffré disponible sur votre ordinateur; on notera par la suite *adresse 1* le chemin qui permet d'accéder à ce fichier
2. Disposer de la clef de déchiffrement du canal numérique qui a été utilisé pour recevoir le fichier enregistré dans *adresse 1*
3. Installer *java* sur votre poste de travail
4. Installer l'outil de déchiffrement mis à disposition par Enedis et paramétrer cet outil pour son utilisation
5. Lancer cet outil pour déchiffrer le fichier

2.1. Pré-requis

2.1.1. Clé de chiffrement

Les fichiers de données sont envoyés sur un canal numérique que vous avez paramétré dans votre compte client entreprise - https://mon-compte-entreprise.enedis.fr/.

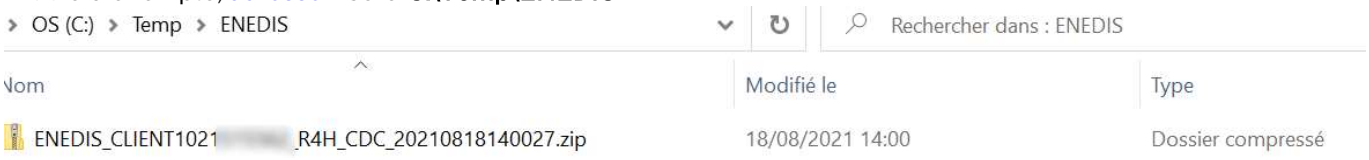
La clé de chiffrement est à récupérer sur le canal numérique dans le compte client entreprise :



Copier cette clé dans un document afin de pouvoir l'utiliser ultérieurement dans cette procédure.

2.1.2. Stockage des fichiers reçus sur le canal numérique

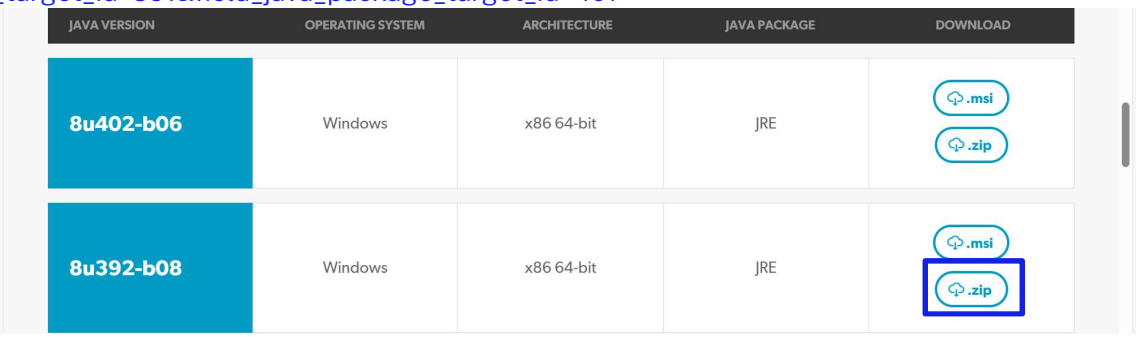
Placer dans un répertoire les fichiers chiffrés reçus d'Enedis. Le chemin d'accès à ces fichiers sera appelé *adresse 1*.
A titre d'exemple, *adresse 1* sera **C:\Temp\ENEDIS**.



2.2. Mode opératoire

2.2.1. Installer Java sur votre poste de travail

Pour installer *Java* sur son poste, il est nécessaire de télécharger le JRE zippé *x86 64-bit* via le lien open source :
https://www.openlogic.com/openjdk-downloads?field_java_parent_version_target_id=416&field_operating_system_target_id=436&field_architecture_target_id=391&field_java_package_target_id=401



Sur la ligne correspondant au JRE zippé x86 64-bit, double-cliquez sur *.zip*
Le fichier se télécharge sur votre ordinateur.

Ouvrez le dossier *Téléchargements* de votre ordinateur.
Pour dézipper un fichier, il faut « Cliquer-droit » sur le fichier téléchargé ⇨ *Extraire tout* ⇨ Saisir le répertoire où vous souhaitez stocker *Java*. Le chemin d'accès de ce répertoire est à conserver pour les prochaines étapes : ce chemin sera appelé *adresse 2*.
A titre d'exemple, adresse 2 sera **C:\Users\Desktop\openlogic-openjdk-jre-8u362-b09-windows-64** .
[Version JAVA pour l'exemple : **JAVA 8, 8u362-b09, JRE, version zip**]

2.2.2. Installer et préparer l'outil de déchiffrement

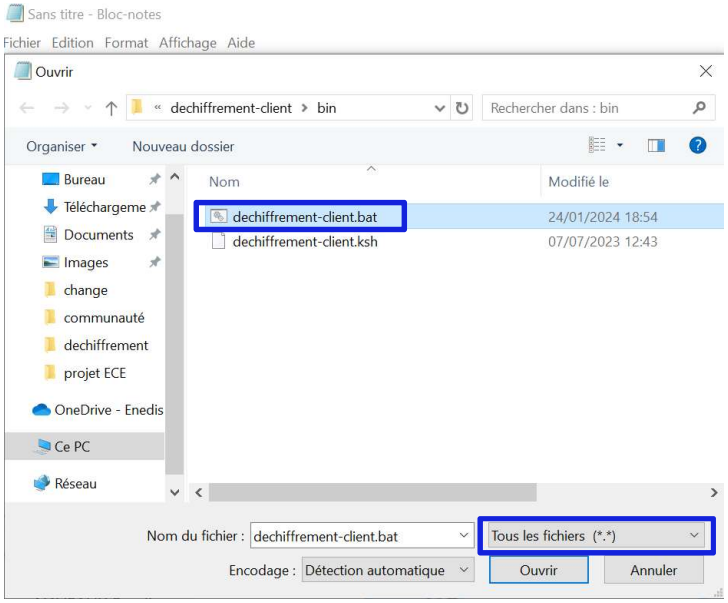
Enedis met à disposition des clients un outil de déchiffrement permettant de faciliter les manipulations réalisées lors de la réception des fichiers chiffrés.
Par défaut, dans l'outil de déchiffrement, le répertoire où les fichiers sont attendus est **C:\Temp\ENEDIS**. Cela peut être paramétré selon votre environnement.

L'étape d'installation de l'outil sur votre poste est identique sous Windows ou sous Linux.
Téléchargez le fichier « dechiffrement-client » sur votre ordinateur.
https://www.enedis.fr/media/3563/download

Ouvrez le dossier *Téléchargements* de votre ordinateur.
Pour dézipper un fichier, il faut « cliquer-droit » sur le fichier téléchargé ⇨ *Extraire tout* ⇨ Saisir le répertoire où vous souhaitez stocker *l'outil Enedis*. Le chemin d'accès de ce répertoire est à conserver pour les prochaines étapes : ce chemin sera appelé *adresse 3* (il peut être identique à *adresse 2*).

Depuis l'*adresse 3*, ouvrir le dossier *Bin* : 2 fichiers sont disponibles.
Le fichier *dechiffrement-client.bat* est le fichier qu'il faut paramétrer. Il faut modifier ce fichier pour l'adapter en fonction des enregistrements réalisés en amont du déchiffrement.

- Ouvrir le bloc note de votre ordinateur.
- Cliquez sur *Fichier* ⇨ *Ouvrir* ⇨ *Tous les fichiers*
- Choisissez le fichier *dechiffrement-client.bat*



- Le fichier s'ouvre en affichant les lignes de commande de l'outil de déchiffrement Enedis.

  2 paramètres sont à modifier (ne pas modifier le reste du document) :

  Pour les PC Windows :

  SET JAVA_HOME : indiquez l'adresse 2 entre les guillemets. Il s'agit du répertoire sur votre PC où vous avez extrait *Java*

  SET REPERTOIRE : indiquez l'adresse 1 entre les guillemets Il s'agit du répertoire sur votre PC où se trouve le fichier à déchiffrer

```
*dechiffrement-client.bat - Bloc-notes
Fichier  Edition  Format  Affichage  Aide
@echo off

setlocal
SET JAVA_HOME="C:\Users\Desktop\openlogic-openjdk-jre-8u362-b09-windows-64"
SET REPERTOIRE="C:\Temp\ENEDIS"

if NOT DEFINED JAVA_HOME goto ERREURJAVA

echo *** Lancement du script de dechiffrement de fichier ***
echo.

set /p _cleAES256= Veuillez saisir votre cle de dechiffrement ?

echo Recuperation des parametres : OK
echo.

echo Recuperation de la liste des fichiers dans le repertoire %REPERTOIRE%
dir /b /a-d %REPERTOIRE% | findstr /v /i "lisible*"  > dechiffrement-client.txt
echo Recuperation OK
echo.

for /f "tokens=*" %%s in (dechiffrement-client.txt) do (
%JAVA_HOME%\bin\java -jar %CD%\..\jar\dechiffrement-client.jar -repertoire=%REPERTOIRE% -fichierChiffre="%%s" -cleAES256=%_cleAES256%
echo.
)

del dechiffrement-client.txt

echo *** Fin du script de dechiffrement de fichier ***
echo off
pause
EXIT /B %ERRORLEVEL%
endlocal
goto :eof
```

Pour les PC Linux :

export JAVA_HOME="/usr/lib/jvm/openlogic-openjdk-jre-8u362-b09-windows-64"

Indiquez l'adresse 2 entre les guillemets. Il s'agit du répertoire sur votre PC où vous avez extrait *Java*

SET REPERTOIRE="C:\Temp\ENEDIS"

Indiquez l'adresse 1 entre les guillemets Il s'agit du répertoire sur votre PC où se trouve le fichier à déchiffrer

- Enregistrer les 2 modifications faites en cliquant sur les touches CTRL+S ou en cliquant sur Fichiers ⇨ Enregistrer
- Fermer le Bloc-notes

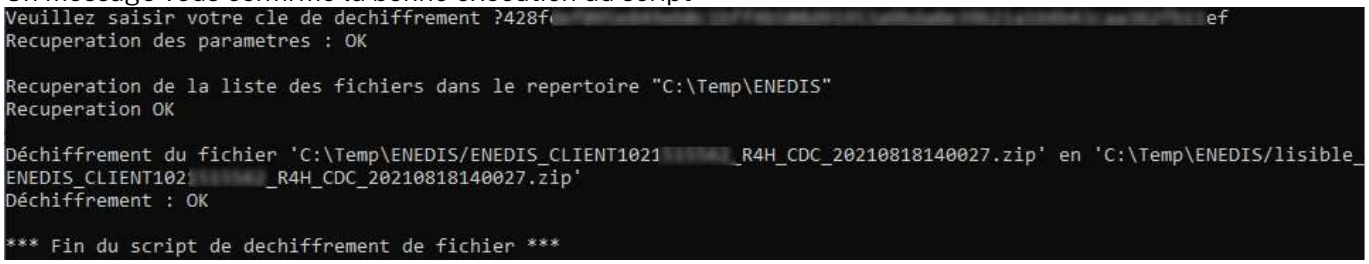
2.2.3. Utilisation de l'outil

Depuis l'*adresse 3*, ouvrir le dossier *Bin*.et « double-cliquez » sur le fichier *dechiffrement-client.bat* pour les PC Windows (ou *dechiffrement-client.ksh* pour les PC Linux)
Une fenêtre s'ouvre et vous demande de saisir la clef de déchiffrement.

```
C:\WINDOWS\system32\cmd.exe
*** Lancement du script de dechiffrement de fichier ***

Veuillez saisir votre cle de dechiffrement ?
```

Saisissez ou copiez la clef de déchiffrement (cf. § 2.1.1 - Clé de chiffrement) et appuyez sur Entrer.
Un message vous confirme la bonne exécution du script



```
Veuillez saisir votre cle de dechiffrement ?428f[illegible]ef
Recuperation des parametres : OK

Recuperation de la liste des fichiers dans le repertoire "C:\Temp\ENEDIS"
Recuperation OK

Déchiffrement du fichier 'C:\Temp\ENEDIS/ENEDIS_CLIENT1021[illegible]_R4H_CDC_20210818140027.zip' en 'C:\Temp\ENEDIS/lisible_
ENEDIS_CLIENT102[illegible]_R4H_CDC_20210818140027.zip'
Déchiffrement : OK

*** Fin du script de dechiffrement de fichier ***
```

Depuis l'*adresse 1*, le fichier déchiffré est disponible avec le préfixe « lisible ».

ANNEXES

Liste des flux chiffrés à destination des clients entreprise

Le tableau suivant liste les flux chiffrés vers les clients entreprise, et indique pour chacun :

- la méthode de chiffrement utilisée par Enedis ; cette méthode pouvant évoluer dans le temps, la date d'application est précisée dans la colonne « A partir du ». Les différentes méthodes de chiffrement sont décrites dans le chapitre 4 « Les différentes méthodes de chiffrement » ;
- la méthode de gestion des clés. Les différentes méthodes de chiffrement sont décrites dans le chapitre 3 « Les différentes méthodes de gestion des clés »

Code Flux	Nom Flux	A partir du	Chiffrement	Gestion des Clés
IFJ	Infra-J	30 Juil. 2019	AES256 IV dyn	Self-Care EC Entreprise
R171	Index quotidiens	Oct. 2019	AES256 IV dyn	Self-Care EC Entreprise
R172	Relevé de glissement	Oct. 2019	AES256 IV dyn	Self-Care EC Entreprise
R4Q, R4H, R4M	Publication Récurrente de Courbe de charge	Oct. 2019	AES256 IV dyn	Self-Care EC Entreprise
R6343	Publication Récurrente de Courbe de charge	Fev. 2023	AES256 IV dyn	Self-Care EC Entreprise
R6419	Publication Récurrente de données Index	Fev. 2023	AES256 IV dyn	Self-Care EC Entreprise

Les différentes méthodes de gestion des clés

A la date de création de ce document, deux méthodes de gestion des clés pour les clients entreprise sont mises en place.

Méthode « Selfcare Espace Client Entreprise (ECE) »

Sur l'Espace client entreprise, la page « Mes canaux de contact » permet à un client entreprise de gérer, en *selfcare*, ses propres canaux de contacts de type « Mail » et « FTP », à la maille d'un SIREN.

Chaque canal de contact dispose de sa propre clé de chiffrement, qui est une clé symétrique de 256 bits qui doit rester secrète, connue seulement d'Enedis (pour chiffrer) et du client (pour déchiffrer).

Cette clé est :

- créée lors de la création d'un nouveau canal de contact dans le compte client entreprise,
- modifiée lors de la modification d'un canal de contact existant : tous les flux reçus suite à cette opération seront chiffrés avec la nouvelle clé.

Pour la lire, il faut, dans le compte client entreprise, cliquer sur l'icône « Clé » du canal de contact : elle est alors affichée sous la forme de 64 caractères hexadécimaux, et un bouton « copier » permet de placer ces 64 caractères dans le « presse-papier » pour la coller dans le système d'information chargé du déchiffrement des publications reçues sur ce canal de contact.
La clé associée à un canal de contact sera utilisée pour chiffrer tous les flux chiffrés publiés sur ce canal de contact. Autrement dit, elle n'est pas associée à un « Code flux », mais au canal lui-même.

Méthode « AES256 IV dyn » : chiffrement AES256 avec IV dynamique

Méthode de chiffrement

L'algorithme AES est utilisé en mode CBC avec une clé de 256 bits et un padding « PKCS5Padding »[1].
Pour chaque chiffrement d'un fichier, un IV (*Initialization Vector*) de 128 bits aléatoires est créé (chaque IV ne servira qu'une seule fois, on parle donc d'IV « dynamique »). La connaissance de l'IV est indispensable pour le déchiffrement, il doit donc être transmis au destinataire. Cette transmission se fait via le fichier chiffré envoyé : les 128 bits de l'IV sont contenus dans les 128 premiers bits (ou 16 octets) du fichier.

Méthode de déchiffrement

La méthode de déchiffrement consiste à utiliser l'algorithme AES256, en mode CBC, en utilisant l'IV lu dans les 128 premiers bits (ou 16 octets) du fichier chiffré reçu. Les données suivantes (après les 128 premiers bits) sont les blocs de données chiffrés en AES.

Exemple de code Java pour le déchiffrement

Si l'outil mentionné précédemment ne convient pas ou ne fonctionne pas, voici un exemple de code JAVA permettant de déchiffrer les fichiers. Cette partie est plus technique que la précédente : JAVA et un IDE sont nécessaires à la bonne compilation du code ci-dessous. Il est aussi nécessaire d'avoir des compétences techniques pour la dérouler.
Le code Java suivant permet de déchiffrer un fichier chiffré en AES256 avec IV (*Initialization Vector*) de 128 bits en en-tête du fichier chiffré.
Il prend 3 paramètres en entrée :

- Le chemin d'accès au fichier chiffré (reçu d'Enedis)
- Le chemin d'accès au fichier déchiffré (qui sera créé)
- La clé symétrique (de chiffrement/déchiffrement) de 256 bits

```
import java.nio.file.Files;
import java.nio.file.Paths; import
java.io.FileOutputStream;
import java.security.SecureRandom;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import javax.crypto.spec.SecretKeySpec;
import
org.apache.commons.codec.binary.Hex;

/**
```

[1] En cas d'anomalie, le padding peut être retiré pour un meilleur fonctionnement

```
         * Dechiffre un stream en AES 256 avec la cle keyString
     *
         * @param encryptedStream = Fichier chiffré
         * @param clearStream = Fichier déchiffré
         * @param keyString = Clé AES256 encodée en Hexadécimal (64 caractères)
         * @throws CryptoException
     */

    public static void decryptStream(final InputStream encryptedStream, final OutputStream
clearStream,
                                     final String keyString) throws CryptoException {
try {
            final byte[] keyBytes = Hex.decodeHex(keyString.toCharArray());
            final byte[] key = new byte[keyBytes.length];
            System.arraycopy(keyBytes, 0, key, 0, keyBytes.length);

            final SecretKey keyValue = new SecretKeySpec(key, "AES");

            final Cipher decryptCipher = Cipher.getInstance("AES/CBC/PKCS5Padding", "SunJCE");
            // Lecture de l'IV depuis le 1er bloc du fichier d'entrée
byte[] iv = new byte[AES256EncryptionUtilDyn.BLOCK_SIZE];
encryptedStream.read(iv, 0, iv.length);
            IvParameterSpec ivspec = new IvParameterSpec(iv);

            decryptCipher.init(Cipher.DECRYPT_MODE, keyValue, ivspec);

            final byte[] buffer = new byte[1024];
int noBytes = 0;
            final   byte[]       cipherBlock   =      new
byte[decryptCipher.getOutputSize(buffer.length)];
            while ((noBytes = encryptedStream.read(buffer)) != -1) {                  final int
cipherBytes = decryptCipher.update(buffer, 0, noBytes, cipherBlock);
                clearStream.write(cipherBlock, 0, cipherBytes);
}

            final int cipherBytes = decryptCipher.doFinal(cipherBlock, 0);
clearStream.write(cipherBlock, 0, cipherBytes);
        } catch (final Exception e) {
            throw new CryptoException("Erreur lors du déchiffrement des données", e);
        } finally {
try {
                if (encryptedStream != null) {
encryptedStream.close();
                }
                if (clearStream != null) {
clearStream.flush();                     clearStream.close();
                }
            } catch (final IOException localIOException) {
}
        }
    }
```