

# Power Week 2025

#pw2025

18 - 19 - 20 novembre 2025

IBM Innovation Studio Paris

**S51 : XML Signature - Principes et mise en œuvre sur IBM i**

19 novembre 14:45 - 15:45

Florian GRADOT  
GAIA MINI SYSTEMES  
florian.gradot@gaia.fr

IBM

common  
FRANCE

# Présentation

## Florian GRADOT

IBM i depuis 2009  
Consultant IBM i



## GAIA / VOLUBIS

Formation (débutant, perfectionnement)  
Expertise IBM i  
Centre de Services



# Sommaire

- Introduction et contexte
- Concepts fondamentaux
- Exemple de mise en œuvre étape par étape
- Conclusion

Power Week

18 -19 - 20 novembre  
2025



# Introduction

# Introduction

- Pourquoi signer des XML ?
  - **Authenticité** : prouver l'émetteur
  - **Intégrité** : détecter les altérations
  - **Non-répudiation** : éviter le déni d'envoi

# Rappels

- Les clés (couple)
  - Clé privée
    - Sert à signer ou déchiffrer
    - Gardée secrète, connue uniquement par le système
  - Clé publique
    - Sert à vérifier une signature ou chiffrer un message
    - Partagée librement avec tout le monde

A ne jamais stocker ensemble !

# Rappels

- Le certificat (Carte d'identité de la clé publique )
  - Il contient :
    - La clé publique
    - L'identité du propriétaire (nom, organisation...)
    - La signature d'une Autorité de certification (ou auto-signé)
    - Période de validité
  - Sert à prouver que la clé publique appartient bien à l'entité mentionnée dans le certificat, grâce à la signature qui atteste cette liaison.

# Rappels

- Autorité de certification (CA)
  - Organisme (public ou privé)
    - Vérifie l'identité d'une personne ou d'une entreprise
    - Valide sa clé publique
    - Produit un certificat signé
  - Sa signature permet de dire :
    - J'atteste que cette clé publique appartient vraiment à cette personne / entreprise



Power Week

18 -19 - 20 novembre  
2025

**IBM**  
*common*  
FRANCE

**IBM**

# Concepts

# Signature numérique vs chiffrement

## ■ Chiffrement

- Sert à cacher un message pour que seul le destinataire puisse le lire
  - L'expéditeur chiffre le message avec une clé
  - Le destinataire déchiffre avec la clé correspondante

## ■ Signature numérique

- Prouve qui a signé et garantit qu'il n'a pas été modifié
  - Signataire crée une empreinte (hash) du document
  - Il signe l'empreinte avec sa clé privée
  - Toute personne peut vérifier avec sa clé publique

# Signature numérique vs chiffrement

- Le chiffrement protège le contenu
- La signature numérique prouve qui a créé ou envoyé le document

Fonction	Signature numérique	Chiffrement
Confidentialité	Non	Oui
Authenticité	Oui	Non
Intégrité	Oui	Non
Non-répudiation	Oui	Non
Lisibilité du message	Visible	Illisible

# XML Signature (XMLDSIG)

- Norme du W3C (<https://www.w3.org/TR/xmlsig-core/>)
- Éléments clés : <Signature>, <SignedInfo>, <SignatureValue>, <KeyInfo>

```
1 <Signature>
2   <SignedInfo>
3     <CanonicalizationMethod/>
4     <SignatureMethod/>
5     <Reference URI="">
6       <DigestMethod/>
7       <DigestValue/>
8     </Reference>
9   </SignedInfo>
10  <SignatureValue>...</SignatureValue>
11  <KeyInfo>
12    <X509Data>...</X509Data>
13  </KeyInfo>
14 </Signature>
```

# Types de signature

- Signature enveloppée
  - Elle est incluse dans le document XML qu'elle signe.

```
<Document>
  <Data>Exemple de données</Data>
  <Signature>
    <SignedInfo>...</SignedInfo>
    <SignatureValue>...</SignatureValue>
  </Signature>
</Document>
```

# Types de signature

- Signature enveloppante
  - Elle englobe les données qu'elle signe.

```
<Signature>
  <SignedInfo>...</SignedInfo>
  <SignatureValue>...</SignatureValue>
  <Object>
    <Data>Exemple de données</Data>
  </Object>
</Signature>
```

# Types de signature

- Signature détachée
  - Elle est stockée séparément du document qu'elle signe.

```
<Signature>
  <SignedInfo>
    <Reference URI="http://exemple.com/data.xml" />
  </SignedInfo>
  <SignatureValue>...</SignatureValue>
</Signature>
```

# Canonisation C14N

- Deux documents XML peuvent sembler différents mais représenter la même information.

```
<person first="Power" last="Week"/>  
<person last="Week" first="Power"></person>
```

- La canonisation convertit des documents en une version unique et standard.
  - Supprime les variables non significatives (espaces, ordre des attributs ..)
  - Uniformiser les noms et encodages des caractères
  - <https://www.w3.org/TR/2001/REC-xml-c14n-20010315>



# Canonisation exclusive

- Variante de la canonisation C14N
- Elle ne propage pas les déclarations de namespaces qui ne sont pas utilisées par l'élément canonisé.
- <https://www.w3.org/TR/xml-exc-c14n/>

```
<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">  
  <xs:element xmlns:xs="http://www.w3.org/2001/XMLSchema">Hello</xs:element>  
</soap:Envelope>
```

# Algorithmes et recommandations

- Hachage :
  - Transformer n'importe quelle donnée en empreinte numérique fixe
  - Un caractère qui change et le résultat sera différent
  - SHA-1, SHA256, SHA-3
  - Recommandations : SHA256

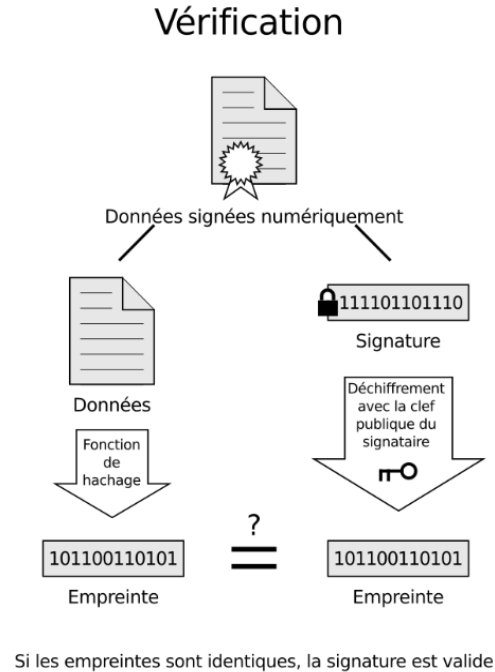
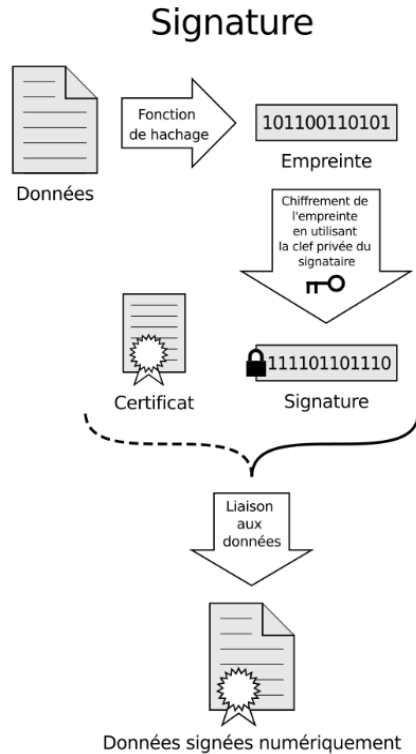
```
<DigestMethod Algorithm="http://www.w3.org/2001/04/xm1enc#sha256" />
```

# Algorithmes et recommandations

- Signature :
  - RSA-SHA1 → RSA pour la signature + SHA-1 pour le hash (moins sûr aujourd'hui)
  - RSA-SHA256 → RSA + SHA-256 (standard actuel et sécurisé)

```
<CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />  
<SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" />
```

# Schéma explicatif



Power Week

18 -19 - 20 novembre  
2025



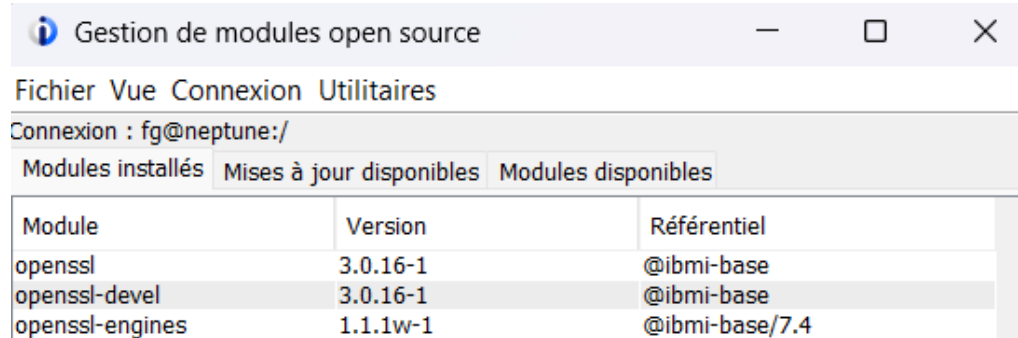
# Cas pratique

# Cas pratique

- A partir des commandes OpenSource
  - Openssl
  - xmllint
- Génération de clé privée / publique manuellement pour le cas pratique

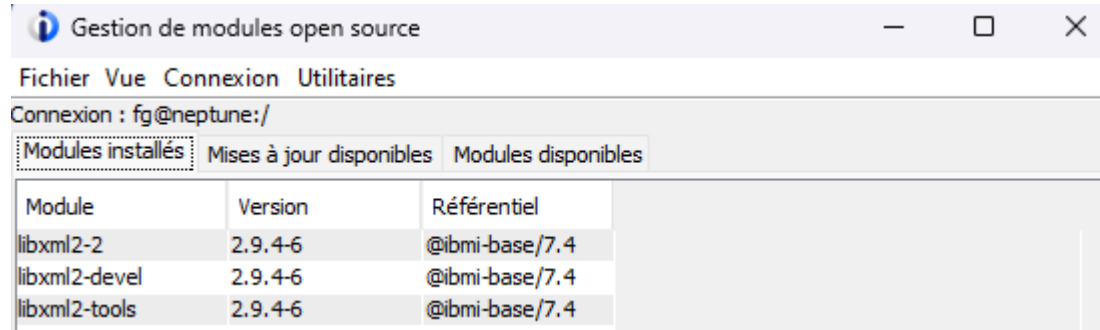
# Pré-requis

- Avoir les modules OpenSource installés



The screenshot shows a window titled 'Gestion de modules open source' with a menu bar containing 'Fichier', 'Vue', 'Connexion', and 'Utilitaires'. Below the menu bar, it says 'Connexion : fg@neptune:/' and has three tabs: 'Modules installés' (selected), 'Mises à jour disponibles', and 'Modules disponibles'. A table lists the installed modules.

Module	Version	Référentiel
openssl	3.0.16-1	@ibmi-base
openssl-devel	3.0.16-1	@ibmi-base
openssl-engines	1.1.1w-1	@ibmi-base/7.4



The screenshot shows the same window as above, but with the 'Modules disponibles' tab selected. The table lists the available modules.

Module	Version	Référentiel
libxml2-2	2.9.4-6	@ibmi-base/7.4
libxml2-devel	2.9.4-6	@ibmi-base/7.4
libxml2-tools	2.9.4-6	@ibmi-base/7.4

# Fichiers d'exemple utilisés

- Key.pem = clé privée
- Key.pub = clé publique
- pw2025.xml = XML à signer
- pw2025\_c14n.xml = XML canonisé
- hash.bin = hash du fichier xml
- pw2025.signature = signature du fichier



# Etape 1 : Génération de la clé privée

## ■ Commande :

`/QOpenSys/pkgsrc/bin/openssl genrsa -out key.pem 4096`

- **genrsa** : commande pour générer une clé RSA privée
- **-out key.pem** : fichier de sortie
- **4096** : taille de la clé en bits

```
> cd /home/fg/PW2025
$
> /QOpenSys/pkgsrc/bin/openssl genrsa -out key.pem 4096
$
```

Répertoire

/home/FG/PW2025

Icône	Nom	Taille (Ko)	Dernière modification
	key.pem		3 14 novembre 2025 à 16:14:28 UTC+1

## Etape 2 : Extraction clé publique


- Commande :


```
/QOpenSys/pkgsrc/bin/openssl rsa -in key.pem -pubout > key.pub
```


- RSA : utilitaire OpenSSL pour manipuler les clés RSA
- -in key.pem : clé privée en entrée
- -pubout > key.pub : clé publique en sortie

```
> /QOpenSys/pkgsrc/bin/openssl rsa -in key.pem -pubout > key.pub  
writing RSA key  
$
```

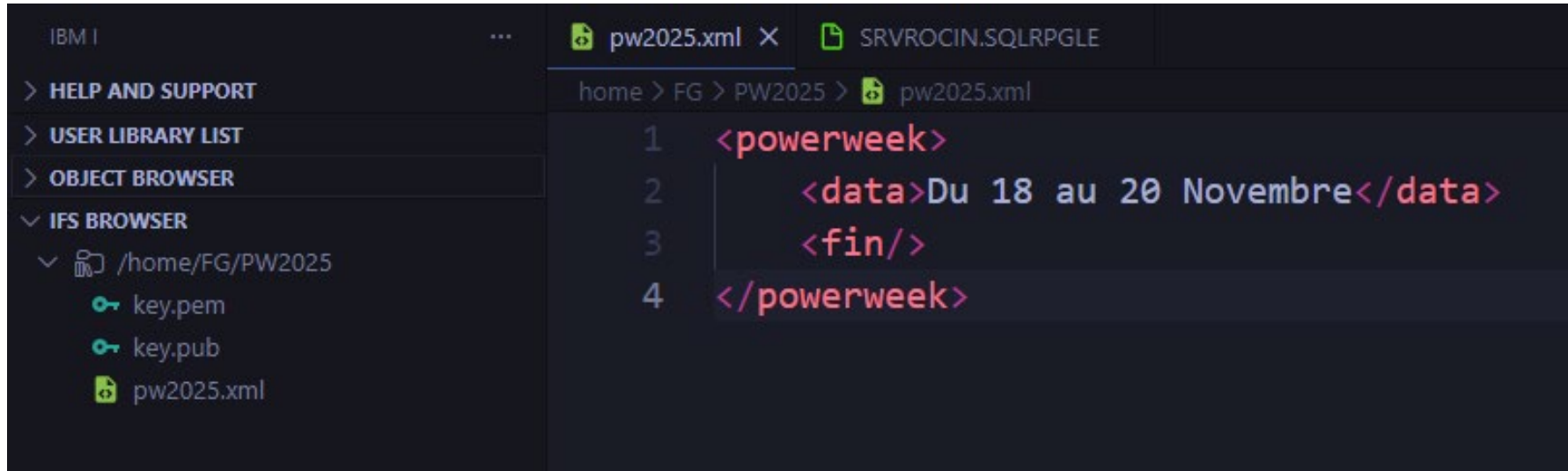
### IFS BROWSER

✓  /home/FG/PW2025

 key.pem

 key.pub

## Etape 3 : Déposer le XML à signer



```
IBM I ... pw2025.xml X SRVROCIN.SQLRPGLE
> HELP AND SUPPORT
> USER LIBRARY LIST
> OBJECT BROWSER
v IFS BROWSER
  v /home/FG/PW2025
    key.pem
    key.pub
    pw2025.xml

home > FG > PW2025 > pw2025.xml
1  <powerweek>
2      <data>Du 18 au 20 Novembre</data>
3      <fin/>
4  </powerweek>
```

## Etape 4 : Canoniser le XML

- Commande :

`/QOpenSys/pkgsrc/bin/xmllint --exc-c14n pw2025.xml > pw2025_c14n.xml`

- `--exc-c14n` : canonisation exclusive
- `pw2025.xml` : fichier à canoniser
- `pw2025_c14n.xml` : fichier canonisé

```
> /QOpenSys/pkgsrc/bin/xmllint --exc-c14n pw2025.xml
<powerweek>
  <data>Du 18 au 20 Novembre</data>
  <fin></fin>
</powerweek>$
```

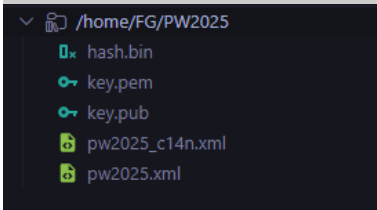
# Etape 5 : Générer le hash

## ■ Commande :

`/QOpenSys/pkgsrc/bin/openssl dgst -binary -sha256 pw2025_c14n.xml > hash.bin`

- **Dgst** : commande pour calculer le hash (digest)
- **-binary** : sortie au format binaire
- **-sha256** : Algorithme de hash
- **pw2025\_c14n.xml** : Fichier dont on calcule le hash
- **> hash.bin** : Hash au format binaire

```
> /QOpenSys/pkgsrc/bin/openssl dgst -binary -sha256 pw2025_c14n.xml > hash.bin
$
```

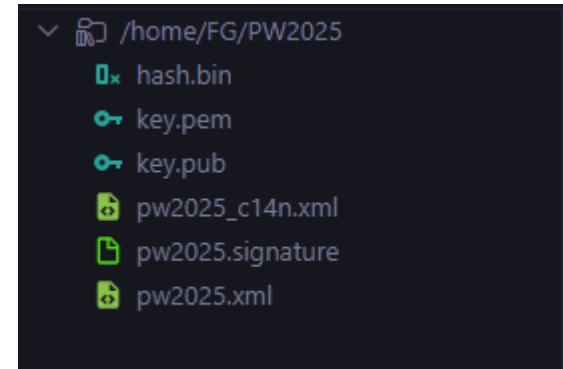


# Etape 6 : Générer la signature

## ■ Commande :

```
/QOpenSys/pkgsrc/bin/openssl pkeyutl -sign -in hash.bin -inkey key.pem  
-keyform PEM -pkeyopt digest:sha256 -out pw2025.signature
```

- `openssl pkeyutl -sign` : signer des données
- `-in hash.bin` : hash du fichier xml
- `-inkey key.pem` : clé privée
- `-keyform PEM` : Format de la clé privée
- `-pkeyopt digest:sha256` : Algorithme de hash
- `-out pw2025.signature` : Signature en sortie



# Etape 7 : Contrôler la signature

- Commande :

```
/QOpenSys/pkgsrc/bin/openssl dgst -verify key.pub -keyform PEM -sha256  
-signature pw2025.signature -binary pw2025_c14n.xml
```

- **Dgst** : Manipuler les digests (hash)
- **-verify key.pub** : Clé publique pour vérifier la signature
- **-keyform PEM** : format de la clé
- **-sha256 -binary** : Hash utilisé pour générer le hash avant signature + format
- **-signature pw2025.signature** : Fichier contenant la signature
- **pw2025\_c14n.xml** : fichier original

```
> /QOpenSys/pkgsrc/bin/openssl dgst -verify key.pub -keyform PEM -sha256 -signature pw2025.signature -binary pw2025_c14n.xml  
Verified OK  
$
```

# Pour une signature enveloppée

- Les étapes :
  - Calculer le hash du XML
  - L'ajouter dans la balise <SignedInfo> + renseigner la référence
  - Calculer le hash et signer la balise <SignedInfo>
  - Ajouter la signature dans la balise <SignatureValue>
  - Incorporer tout le bloc <Signature> dans le XML d'origine



# Exemple résultat final

```
1  <Envelope xmlns="http://example.org/envelope">
2    <Body>
3      <powerweek Id="_3B973EB7-7EAD-4A0E-AE16-F0ED681B9928">
4        <data>Du 18 au 20 Novembre</data>
5        <fin/>
6      </powerweek>
7    </Body>
8    <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
9      <SignedInfo>
10        <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
11        <SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256" />
12        <Reference URI="#_3B973EB7-7EAD-4A0E-AE16-F0ED681B9928">
13          <Transforms>
14            <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
15            <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" />
16          </Transforms>
17          <DigestMethod Algorithm="http://www.w3.org/2001/04/xmldsig#sha256" />
18          <DigestValue>47DEQpj8HBSa+/T+mWp5JCeyERkm5NMpJWZG3hSuFU=</DigestValue>
19        </Reference>
20      </SignedInfo>
21      <SignatureValue>
22        TSQUoVrQ0kg1eiltNwIhKPrIdssuedWjYNJlXvfQqW2EKk3X37X862SCfrz7v8IYJ7OorWwlFpGDStJDSR6sa0
23        ScqSvmesCr+FEq+U6zegR9nH0LDIAL8Rvc/y7U9kZrE4fHqEiLyfpmzJyPmWUT9Uta14nPJYs13cmdThHB8Bs=
24      </SignatureValue>
25    </Signature>
26  </Envelope>
```

Power Week

18 -19 - 20 novembre  
2025



# Conclusion

# Conclusion

- Il est possible sur l'IBM i de signer des XML grâce à l'OpenSource
- Bien faire attention au rajout de la signature dans le XML
- Attention au nombre de fichiers générés
- Alternative:
  - Utiliser les API IBM : Qc3EncryptData et Qc3DecryptData
  - JAVA ? PHP ? Autre ?

MERC

The word "MERC" is displayed in large, bold, white capital letters with a subtle drop shadow. Each letter serves as a frame for a portrait of a diverse professional. The 'M' features a woman with long dark hair wearing a green top. The 'E' features a man with a beard wearing a green patterned shirt. The 'R' features a woman with dark hair wearing a light blue top, with her hands clasped in front of her. The 'C' features a man in a blue suit and yellow tie. The final 'C' features a man with glasses wearing a blue shirt.