AUXILIEN Katia, SALA-MOCHIZUKI Yûki, JACQUEMIN Paul, TREMOULET-BRETON Loan, VACHALDE Rémi

# Tests : Défaillances.

Rapport de tests.

## Le package SigPack

Il y avait des défaillances au niveau de la vérification de la signature du vendeur à la fin du sprint 1. Elles étaient dues à deux mauvaises utilisations des classes du package `SigPack` dans l'implémentation.

Dans le cas de `SigPack_EncOffer` , il fallait cast `byte[]` lors de l'appel à `getObject()` car à la déclaration object était déjà sous la forme de byte[]. L'erreur était qu'on convertissait `object` à nouveau en `byte[]` avec `SignatureUtils.objectToArrayByte(...)`.

```
auxilienk *
@Test
void testsVerifySignatureWithOneOffer() throws Exception {

    classUnderTest = new SigPack_EncOffer(encPrice,signedPrice,signaturePublicKeyFromKeyStore, bidId: "0");
    assertAll(
        () -> assertTrue(SignatureUtil.verifyDataSignature(encPrice,signedPrice,signaturePublicKeyFromKeyStore)),
        () -> assertFalse(SignatureUtil.verifyDataSignature(SignatureUtil.objectToArrayByte((Object) encPrice),signedPrice,signaturePublicKeyFromKeyStore)),
        () -> assertTrue(SignatureUtil.verifyDataSignature((byte[]) classUnderTest.getObject(),classUnderTest.getObjectSigned(), classUnderTest.getSignaturePubKey())),
        () -> assertFalse(SignatureUtil.verifyDataSignature(SignatureUtil.objectToArrayByte(classUnderTest.getObject()),classUnderTest.getObjectSigned(),
                classUnderTest.getSignaturePubKey()))
    );

}
```

Concernant `SigPack_EncOffersProductTests`, l'erreur d'implémentation était l'inverse du cas précédent. En voulant corriger l'erreur précédente, j'ai propagé une nouvelle erreur dans les appels de `getObject()`. L'erreur a été corrigée en effectuant le cast uniquement pour les classes de `SigPack` qui déclarent leur `object` en `byte[]`.

```
@Test
void testsVerifySignatureWithOneOffer() throws Exception {

    classUnderTest = new SigPack_EncOffersProduct(b,bsigned,signaturePublicKeyFromKeyStore3,setSigPackEncOffer);

    assertAll(
            () -> assertTrue(SignatureUtil.verifyDataSignature(SignatureUtil.objectToArrayByte(b),bsigned,signaturePublicKeyFromKeyStore3)),
            () -> assertTrue(SignatureUtil.verifyDataSignature(SignatureUtil.objectToArrayByte(classUnderTest.getObject()),
                    classUnderTest.getObjectSigned(), classUnderTest.getSignaturePubKey()))
    );

}
```

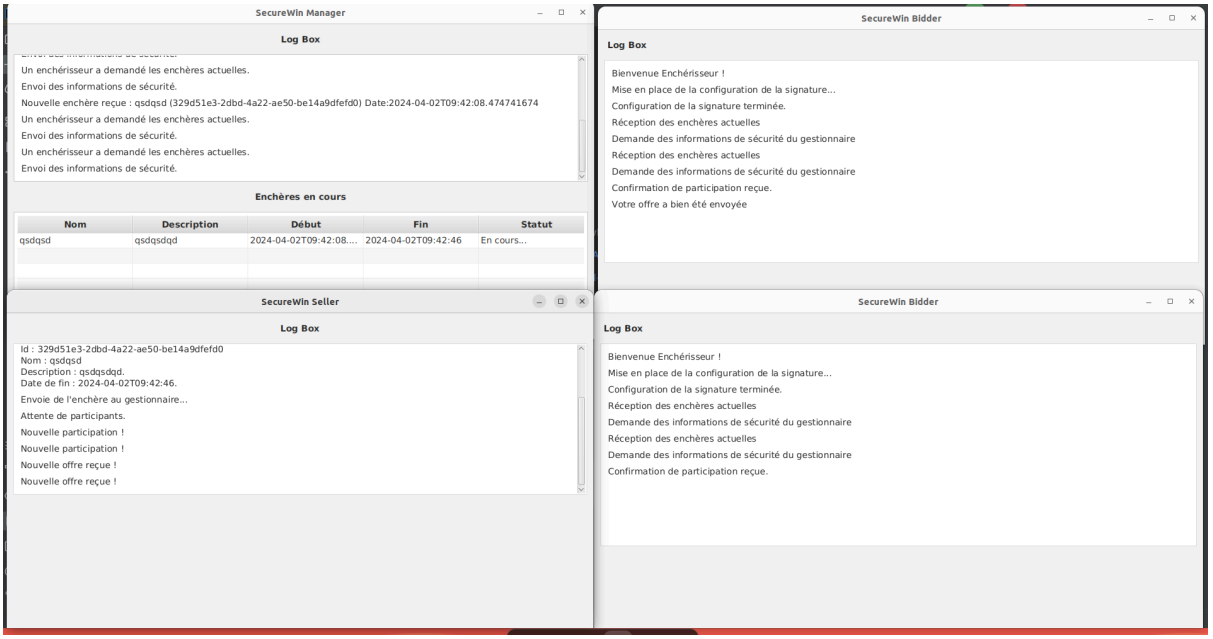
Enfin, il y avait une erreur dans l'utilisation de `SigPack_Results` pour la vérification de la signature du vendeur par l'enchérisseur. Il ne suffisait pas d'appeler `getObject()` mais d'appeler `getObject()` deux fois sur l'objet renvoyé.

```
auxilienk *
@Test
void testsVerifySignatureWithOneOffer() throws Exception {
    resultsSigned = SignatureUtil.signData(sigPackPriceWin.getObject(),signature2);

    classUnderTest = new SigPack_Results(sigPackPriceWin,resultsSigned,signaturePublicKeyFromKeyStore2, bidId: "0");

    assertAll(
            () -> assertTrue(SignatureUtil.verifyDataSignature(SignatureUtil.objectToArrayByte(sigPackPriceWin.getObject()),resultsSigned,signaturePublicKeyFromKeyStore2)),
            () -> assertTrue(SignatureUtil.verifyDataSignature(SignatureUtil.objectToArrayByte(((SigPack_PriceWin)classUnderTest.getObject()).getObject()),
                    classUnderTest.getObjectSigned(),classUnderTest.getSignaturePubKey())),
            () -> assertFalse(SignatureUtil.verifyDataSignature(SignatureUtil.objectToArrayByte(classUnderTest.getObject()),classUnderTest.getObjectSigned(),classUnderTest.getSignaturePubKey()))
    );
}
```

## Le package network




Lors du développement, nous avons rencontré des problèmes de synchronisation des threads.

Cela était dû à un ajout de code pour attendre qui ne devait pas se trouver là, il a été ajouté par inattention et fatigue.