Cryptographic RBAC Compiler

First Sprint

15/10/18 - 28/10/18

FONDAZIONE BRUNO KESSLER

Layout of the slides by Coman Catalin Andrei

Agenda

- Sprint Structure
- Sprint Backlog
- User Stories
- Design
- Implementation
- Testing
- Notes
- Questions

Sprint Structure

- Choose the **Sprint Backlog** from the Product Backlog
- Specify the **Requirements** through user stories
- Create Sequence UML Diagram (**Design**)
- Create tests from user stories (optional)
- **Implementation**
- **Testing**
- **Maintenance** and refactoring
- **Sprint Review**

Sprint Backlog

Sprint Backlog: start by adding **User** and **Role** classes to the system. Generate and distribute **Keys**. No DB (integration with SecurePG will come later). **Tuple** creation and signing process.

Related components:

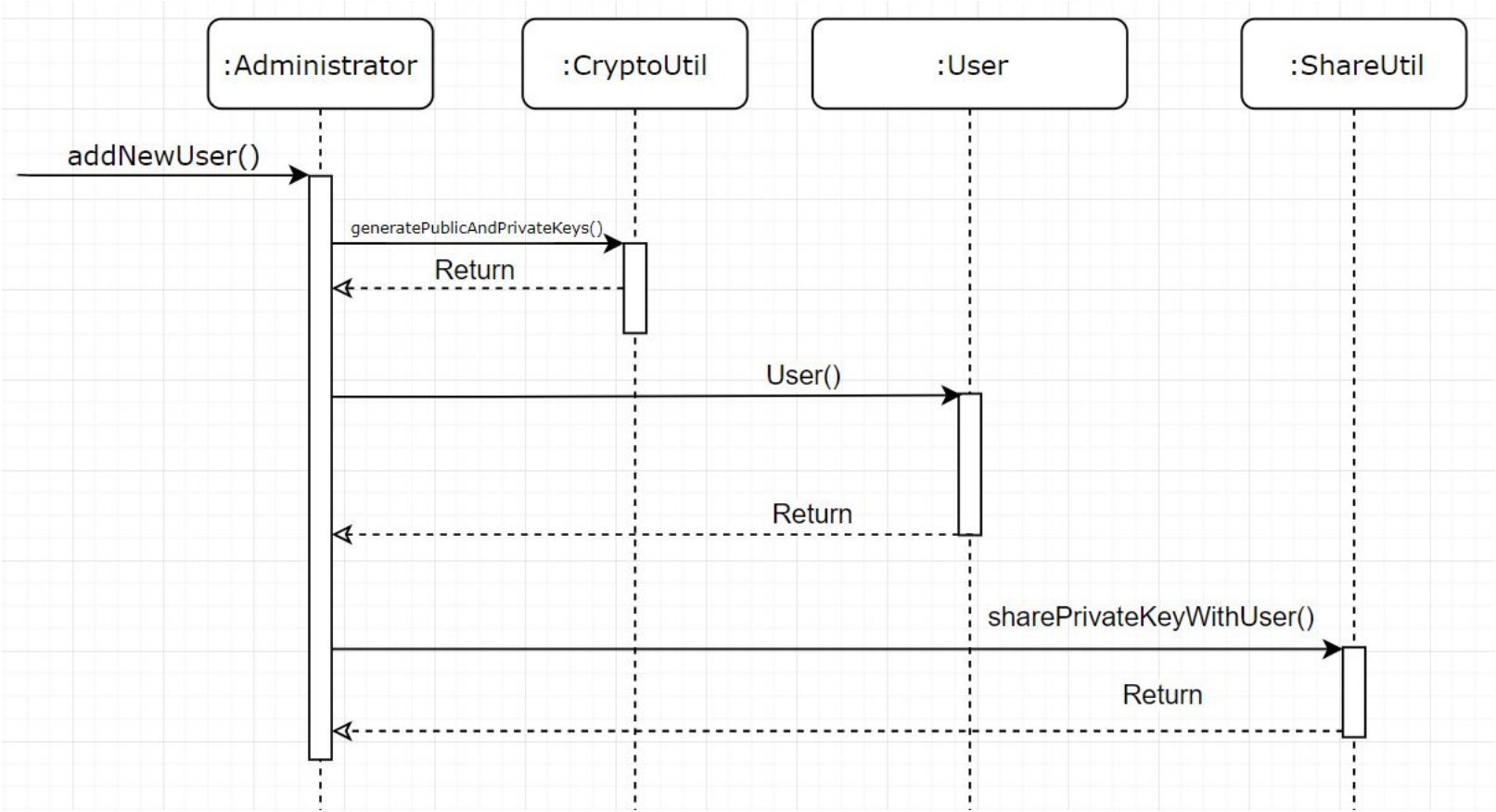
- CryptoUtil (generatePublicAndPrivateKeys, encrypt/decryptKeysWithPKIKey, ...)
- User (getUsername, getEmail, getPublicKey)
- CommandInterpreter (addNewUser, addNewRole)
- Administrator (addNewUser, addNewRole)
- Role (getRoleName, getPublicKey, getRoleVersioNumber)
- TupleAssociableElement
- ShareUtil (shareKeysWithUser)
- Tuple (and subclasses)

1. As a new **User**, I want to ask the administrator to register me into the system in order to get and store my cryptographic keys

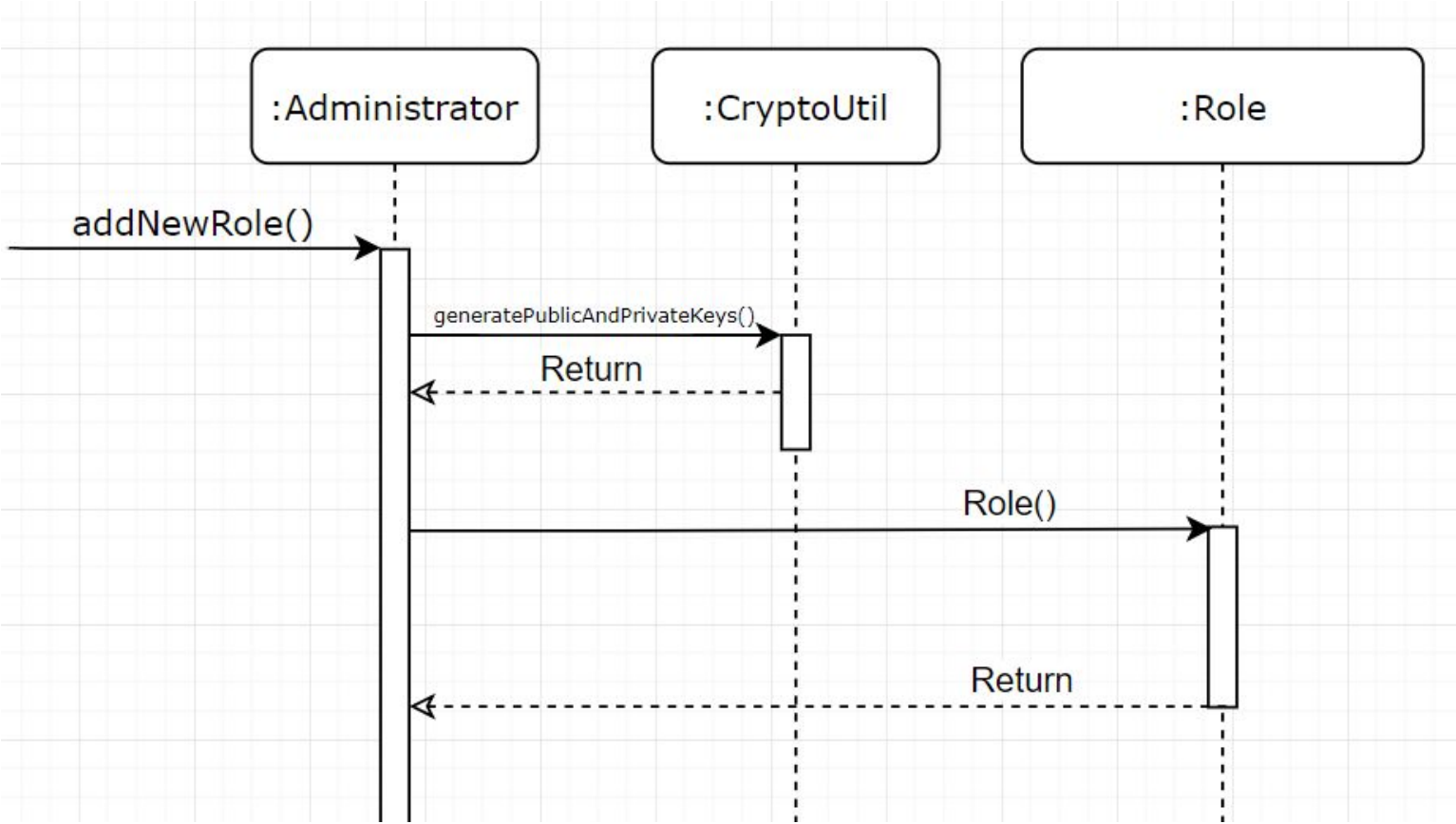
2. As an **Administrator**, I want to instantiate a new Role in order to create its keys.

3. As an **CryptoACActiveElement** (e.g.Role), I want to sign a tuple to later verify it

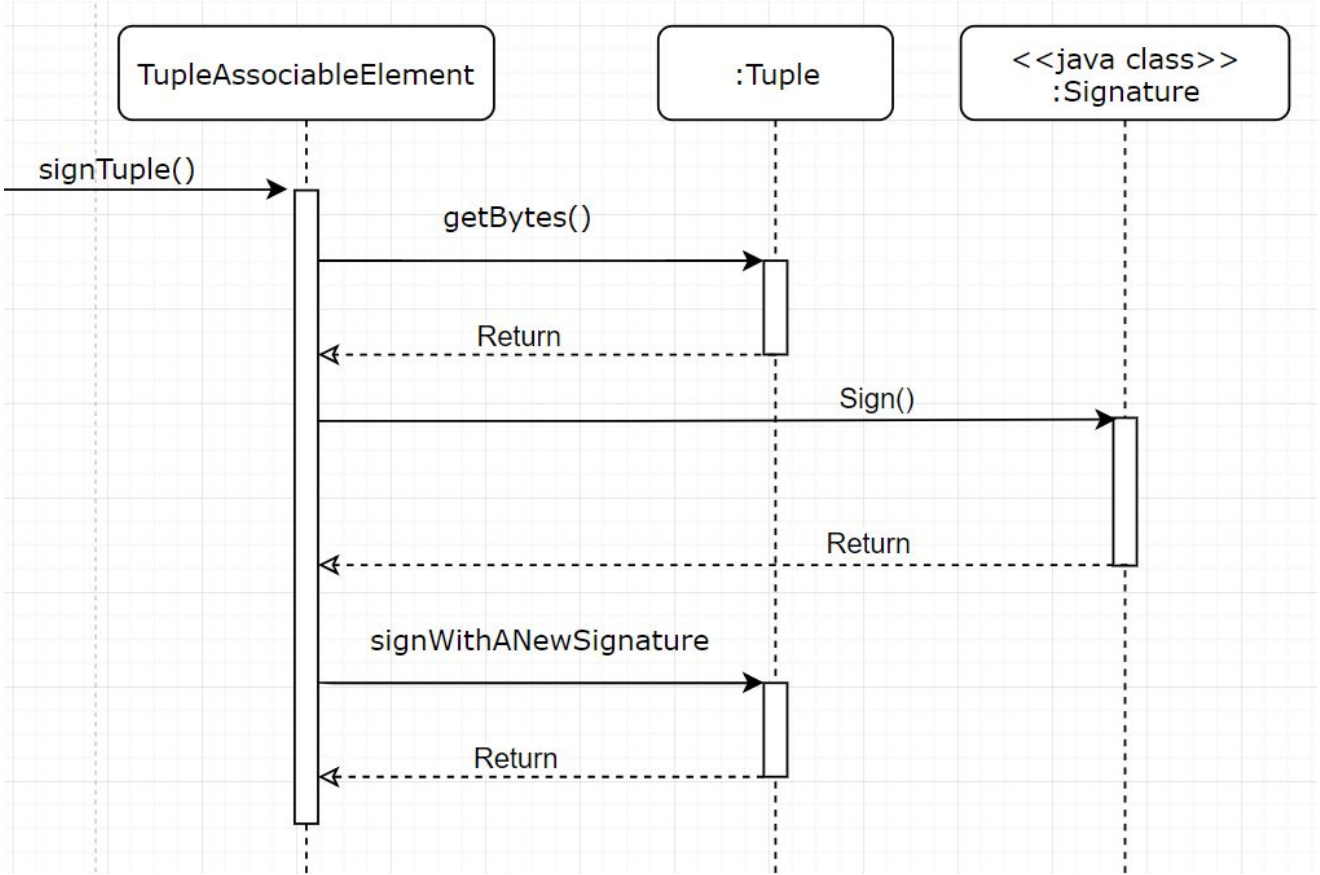
Sequence UML - 1



Sequence UML - 2



Sequence UML - 3



Implementation

Main.java | CryptoUtil.java | CommandInterpreter.java | Administrator.java | Role.java | TupleAssociableElement.java | ShareUtil.java | User.java

Cryptographic-RBAC-Compiler / src

Project

- Cryptographic-RBAC-Compiler ~/Des
  - .idea
  - src
    - eu.fbk
      - main
        - Main
      - object
        - tuple
          - FileTuple
          - PermissionTuple
          - RoleTuple
          - Tuple
        - Role
        - TupleAssociableElement
        - User
      - singleton
        - util
          - CryptoUtil
        - Administrator
        - CommandInterpreter
      - util
        - Permission
        - ShareUtil
        - TupleType

```java
/**...*/
public KeyPair decryptKeysWithPrivateKey (byte [][] keysToDecrypt, PrivateKey decryptingKey)
        throws InvalidKeyException, BadPaddingException, IllegalBlockSizeException, NoSuchAlgorithmException,

    // array of two arrays of bytes
    byte [][] intermediateResult = new byte[2][];


    // ---------    DECRYPTING THE PRIVATE KEY    --------
    // store the encrypted bytes of the private to decrypt
    byte [] inputPrivateKeyInBytes = keysToDecrypt[0];

    // apply the pkiCipher on the private key
    intermediateResult[0] = applyCipherOnBytes (decryptingKey, inputPrivateKeyInBytes, Cipher.DECRYPT_MODE);


    // ---------    DECRYPTING THE PUBLIC KEY    --------
    // store the encrypted bytes of the public to decrypt
    byte [] inputPublicKeyInBytes = keysToDecrypt[1];

    // apply the pkiCipher on the public key
    intermediateResult[1] = applyCipherOnBytes (decryptingKey, inputPublicKeyInBytes, Cipher.DECRYPT_MODE);


    // now rebuild the keys
    KeyFactory kf = KeyFactory.getInstance(this.pkiAlgorithm);
    PrivateKey privateKey = kf.generatePrivate(new PKCS8EncodedKeySpec(intermediateResult[0]));
    PublicKey publicKey = kf.generatePublic(new X509EncodedKeySpec(intermediateResult[1]));

    // finally, return the keys
    return new KeyPair(publicKey, privateKey);
}



/**...*/
private byte[] applyCipherOnBytes(Key PKIkeyToUse, byte[] input, int opmode)
        throws BadPaddingException, IllegalBlockSizeException, InvalidKeyException {
```

Testing

1. Test encrypt and decrypt of keys
2. Test User and Role creation (assert consistency)
3. Test tuple signature and verify process (assert signature is valid)

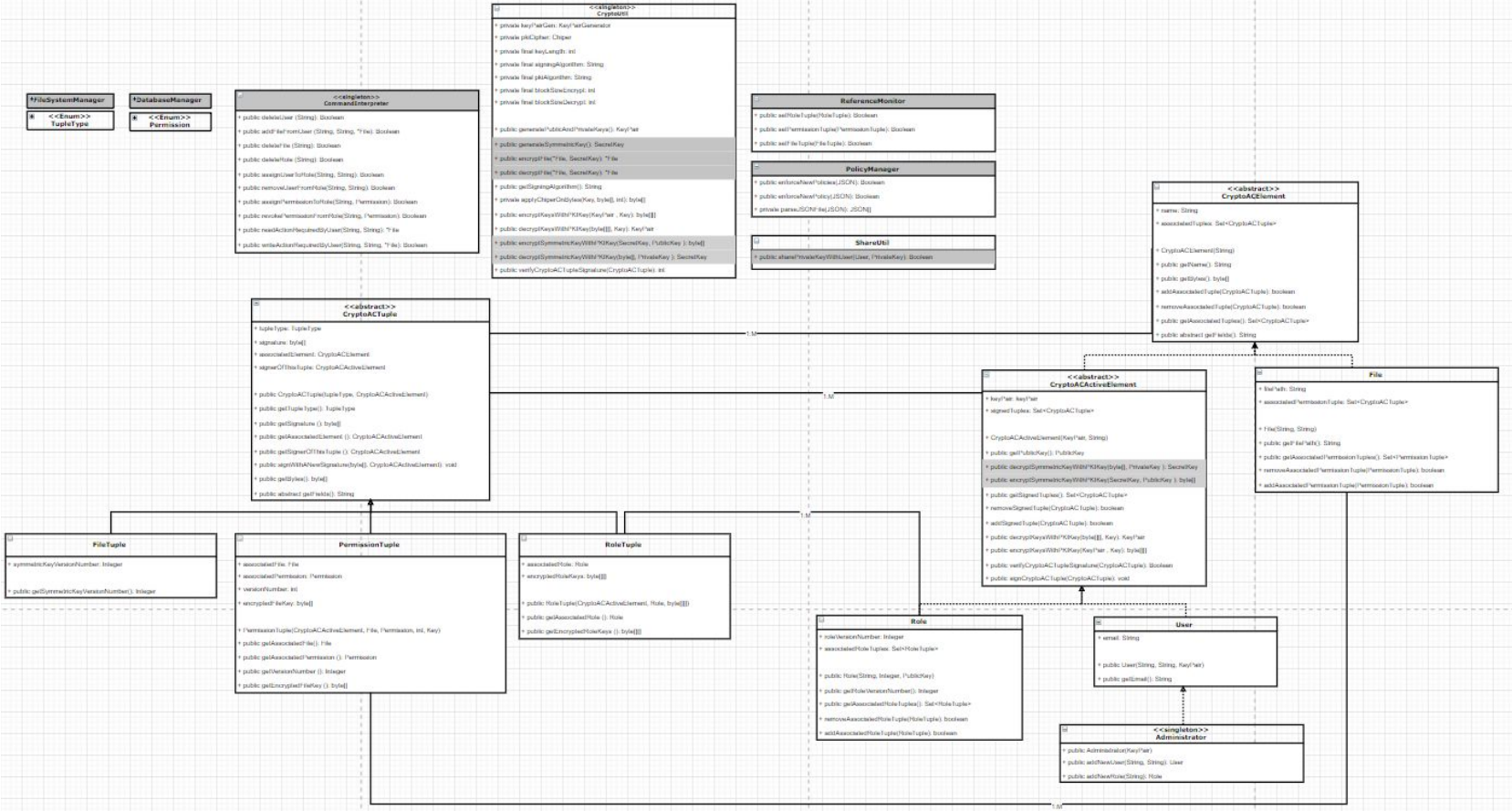
Notes

1. UML class diagram changed

Notes

1. UML class diagram changed
2. Sprint backlog was extended with Tuple objects creation

UML Class Diagram



Questions

- The **`KeyGen`** function requires the **`msk`** parameter (master secret key by the administrator). Why is it necessary?

- Is it preferable to implement the RBAC0 algorithm using **`IBE/IBS`** or using **`PKI`**? And why so? (keep in mind that is a prototype)

Thanks

FONDAZIONE BRUNO KESSLER

Layout of the slides by Coman Catalin Andrei