Approach for decrypt the enrolment packet and store encrypted files in DFS

# Background

Enrolment packets created by the enrolment clients will be periodically uploaded to the server for processing. The packets will be scanned for viruses and after successful scan it will be uploaded to DFS.

The target users are

* Server application which will process the packets
* Administrator of the platform who may need to verify the packets

The key requirements are

* Find out enrollment ids, which are successfully uploaded to DFS and ready for decryption.
* Unzip the packet and gets one encrypted zip file and MetaInfo.json file.
* Decrypt the encrypted zip file and receives a Zip file.
* Unpack the Zip file.
* Store the unpacked files in DFS.
* Updates the status of the packets in the Enrolment Status Table.

The key non-functional requirements are

* Performance: Should be able to support processing multiple packet requests per second.

# Solution

The key solution considerations are -

* Create a batch job “PacketDecryptorJob” to run periodically based on configuration.
* Get the enrollment ids of the packets that are successfully uploaded in DFS.

Service: EnrollmentStatusService.getByStatus(PACKET\_UPLOADED\_TO\_DFS))

* Call the unpack functionality from DFS adapter module which will do the following -
  + Unpack the outer zip file.
  + Store the inner encrypted zip and metainfo file back to DFS.

Service: DFSAdapter.unpackPacket(EnrollmentId))

* The enrollment client will generate asymmetric key (*asymkey*) and use it to encrypt the packet. The *asymkey* will be further be encrypted using the public key of the client and will be sent along with the packet. All public and private key for client ids will be present in server. Now server has to get the private key of the same client id and decrypt the *asymkey* first. After decrypting, it uses same *key* to decrypt the enrollment packet.

TODO: Call the “Key Management” module from core kernel to get private key for given client id. Call “MosipDecryptor” to decrypt *asymkey*. Again, call MosipDecryptor to decrypt inner zip file.

Maven dependency –

<dependency>

<groupId>org.mosip</groupId>

<artifactId>kernel-security</artifactId>

<version>1.0.0-SNAPSHOT</version>

</dependency>

* Once the packet is successfully decrypted, use dfs adapter module to store the encrypted files inside packet back to DFS.

Service: DFSAdapter.storePacket(EnrollmentId, FilePath)

* Update packet status by calling enrollment-status service.
* In case of failure only update enrollment status as “PACKET\_DECRYPTION\_FAILED”

Service: EnrollmentStatusService.updateEnrollmentStatus(enrollmentStatusDto))

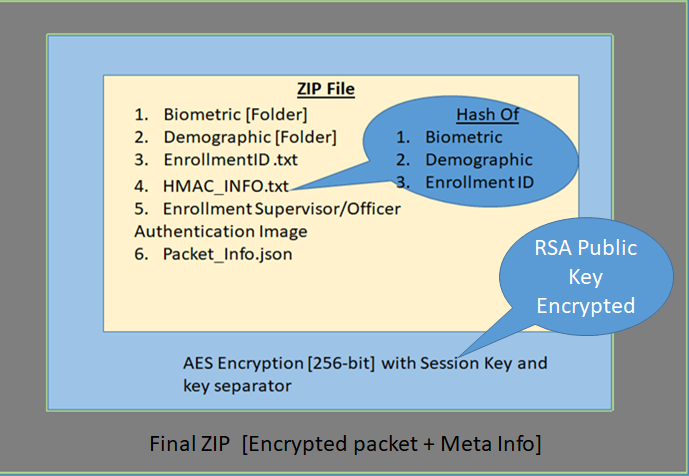
* Send the enrollment id as json format by calling eventbus interface for further processing.

Service: eventBus.send("packet.structure.validation.input", JsonObject);

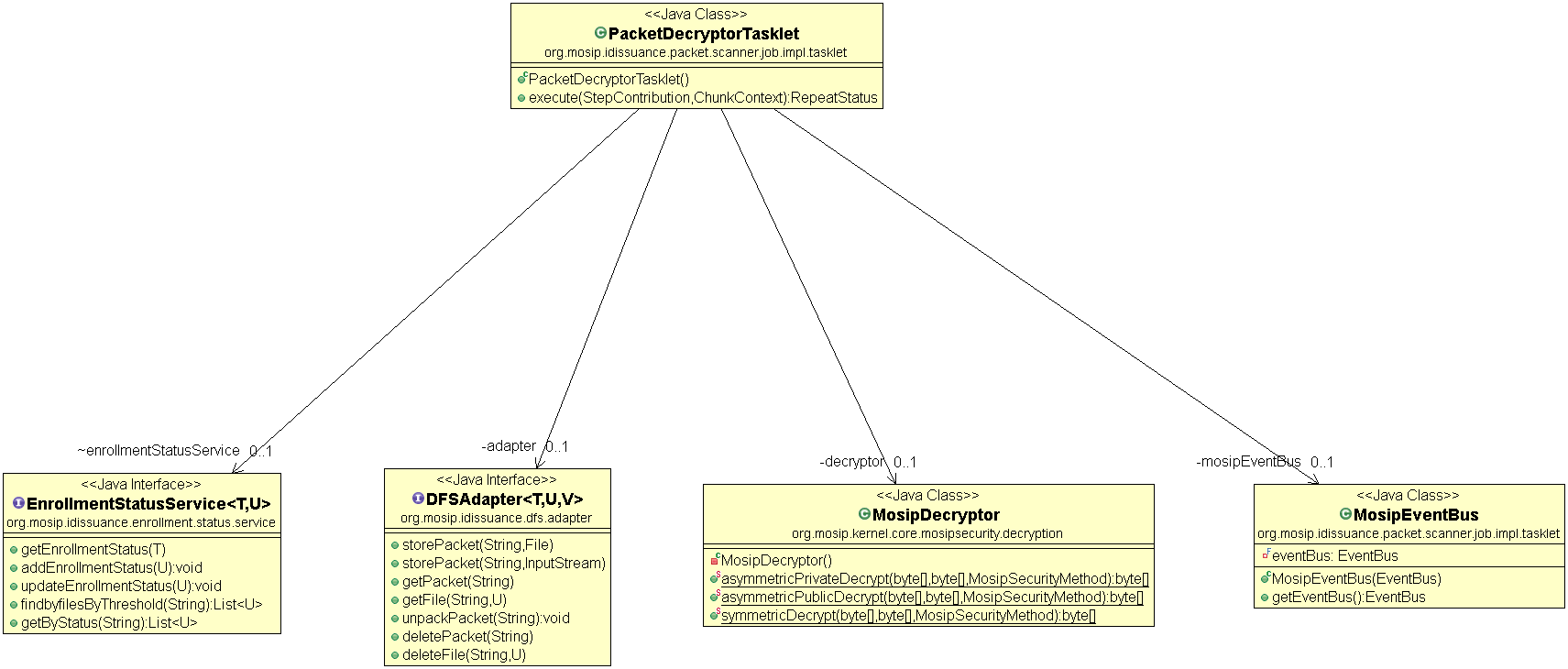
Note: "packet.structure.validation.input" is the camel address where structural validation vertical will listen and pick request for processing.

* Audit the entire transaction.

Enrollment packet structure:

****

Class Diagram:



Sequence Diagram:

