Approach for registration packet validation

# Background

The encrypted packets will be stored in DFS after successful virus scan. A batch job will decrypt packets and store files back to DFS. Next step is packet validation.

The target users are

* Server application, which will store sync information.
* Administrator of the platform who may need to verify the sync info.

The key requirements are

* An eventbus framework which will be responsible for orchestration of the entire flow based on configuration.
* Packet integrity validation by comparing checksum value.
* Validate the list of required files are present in packet before start processing.
* Validate registration machine details, officer details, center details.

The key non-functional requirements are

* Performance: Should validate hundreds of packets per second.

# Solution

The key solution considerations are -

* Registration processor will use event bus to process the packets.
* The batch job ‘packet-decryptor’ will decrypt the packet and send a request to the event bus with the registration id to let the processor would know that packet could be picked for further processing.
* A vertical will be created to process the packets in each stages. Routing request to each vertical is configurable in Registration-processor. If a country wants to control/change the process flow then it should be able to do it. A camel bridge will be created to route request from an address to a vertical and vice versa. For example, Camel will read configuration and send request from camel address to an address where next vertical is listening. Similar way, if a vertical sends response to an address then camel will read configuration and route it to next camel address. The whole process flow of packet should be configurable through camel.



The routing logic will reside in camel and verticals will only consume request from an address and send response to another address. The verticals would be deployed independently and it will not know where the request would go next for processing.

* A ‘packet-validator’ vertical to validate –
  + Required files present inside packet.
  + Integrity of the packet by comparing checksum value.
  + Registration machine details, officer and center details.

1. Validation – 1: Validate Required files present inside packet

The packet decryption job will decrypt packet and store all files inside file system using FileSystemAdapter service. Additionally the service provides a functionality to check if the file is present for a registration id inside DFS. Server will use this functionality to check if all required files are present inside packet to start processing.

Inside packet, client will send PacketMetaInfo.json file where all required file names will be present inside “hashSequence”. Server would read all file names and check if all files are present inside DFS.

If the required files are not present, then send failure response to error queue and update the registration-status table. The client will read packet status and has to resend the packet again.

If the validation is successful then the flow will go to next validation (Integrity of the packet by comparing checksum value).

1. Validation – 2: Integrity of the packet by comparing checksum value

By end of sprint-3 kernel will have HMACGeneration.generatePacketDtoHash() method which takes files and sequence in which HMAC gets generated and in return it gives back the hash. Registration-processor will use same functionality to generate HMAC. The packet will have HMACFile.txt file, which has the generated hash from registration client side. Server will compare generated hash with the hash present in HMACFile.txt to check the integrity of the packet. If the packet is modified during transport the hash will not be same.

If the validation fails then reject the packet, update status in registration-status table and send failure response to error queue.

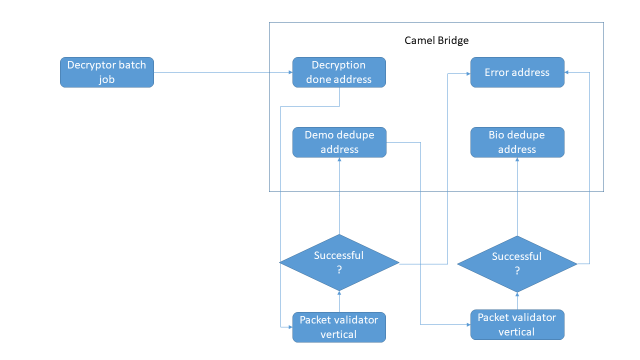
If the validation is successful then go to next validation (Registration machine details, officer and center details).

1. Validation – 3: Registration machine details, officer and center details

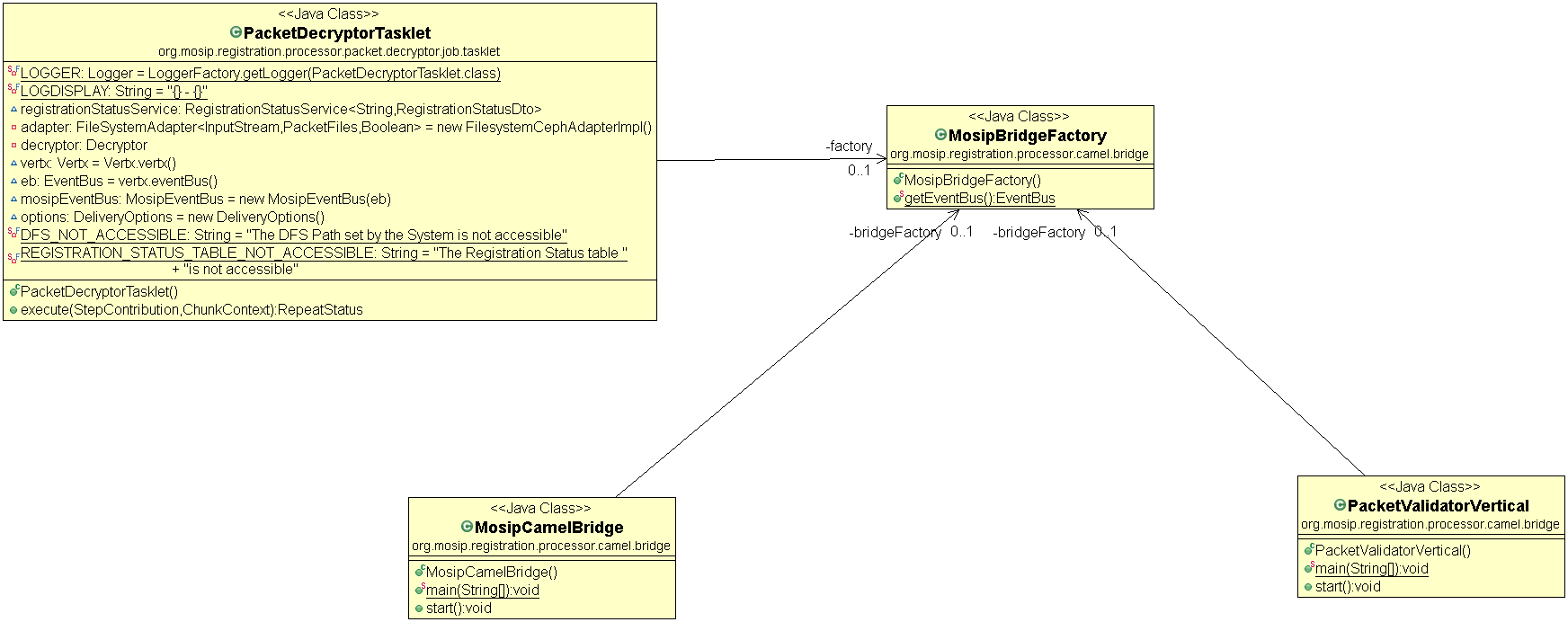
To get the machine details, officer and center details the Registration-processor will call admin service, which will validate and send response. As of now the service is not ready hence a mock service will be created to validate. When the actual service is ready, it will be integrated with code in subsequent sprint. If validation 1 and 2 is successful then the request would come validation-3. Registration-processor will send response to either success or error queue based on the response of admin service.

All above-mentioned three validations should be configurable and if a country wants to skip any validation(s) then it should be able to do it. To achieve this the camel vertical would read the configuration and set validation as true/false in the request header. ‘Packet-validator’ vertical would read the configuration and execute each validation accordingly in the vertical.

Flowchart diagram



Class Diagram:



Sequence Diagram:

