**SA WG2 Meeting #129bis S2-1812226**

**November 26 - 30, 2018, West Palm Beach, USA (revision of S2-18xxxx)**

**Source: Huawei, Hisilicon**

**Title: eSBA: Update of solution 15**

**Document for: Discussion/Approval**

**Agenda Item: 6.19**

**Work Item / Release: FS\_eSBA / Rel-16**

*Abstract of the contribution:* *This contribution propose to remove one EN left at the solution 15.*

# Introduction

There are one EN left at the solution 15 as below,

Editor's note: Consumer receives the response with the Pointer of Producer 2. It is FFS how the consumer handles this mismatch.

The intention of this procedure is to clarify that when the original target service instance is replaced due to different reason, e.g. failure, the service framework can re-direct the message to a different service instance. This is to avoid the tight binding between the consumer and producer.

In S2#129 meeting similar requirement has been agreed as a conclusion we need support, i.e. different procedure can be handled by different service instance. So it can be expected that in those case the consumer shall be able to handle response from the different service instance similar as the scenario described in the procedure. So this requirement need be supported.

From the technical view, it is also possible to transfer the additional information which represent the new producer instance in HTTP/2 protocol, e.g. JSON. And which way is better to transfer that information can be left to stage-3 to determine.

# Conclusion

Per above consideration, it is proposed to remove the EN left at the solution 15.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Start of the first change \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

## 6.15 Solution 15: High reliable deployment via the binding information stored at Framework Function

### 6.15.1 Introduction

This solution is to address the Key Issue 4 and in particular how to maintain the bindings between service consumer and respective service producer to support high reliable deployment.

It is based on architecture defined in 6.14 NF/ Service Set based Service Framework. When one Service Instance communicates with another Service Instance, it includes the binding ID information, which is generated by the service producer. The Service Consumer instance stores the received binding ID until the UE context is released. The binding information, i.e. the binding between the binding ID and service instance, is stored in a new functional module within the Service Framework where the service producer is deployed. When the binding is changed, e.g., the service instance is scaling in/out or failure, the communication peer does not need to be aware. Thus the high reliability can be reached if the service instance to be communicated is replaced, e.g. due to failure.

### 6.15.2 High level description

Similar as the definition of the AMF Instance at Rel-15, it is assumed that the service instance is identified by a Service Set ID and Instance Pointer. When the Service Producer Instance is communicated per the Service Consumer request, the Service Producer Instance provides a binding identifier (i.e. binding ID) and returned it to the Service Consumer. The Service Consumer use the binding ID to identify the Producer Instance to be contacted. Two types of bindings ID are defined:

- Service Set ID based, bind to a service set and is not limited to a dedicated Instance.

- Service Set ID and Instance Pointer based. Depending on the meaning of binding ID, it can be bound to a specific service instance but the service instance can be replaced, or only to one dedicated Instance.

The Service Consumer instance stores the received binding ID until the UE context is released, and includes it in the following request targeted to the same Producer Service. When the message reaches the Unit where the service producer instance is located, it is routed to a service producer instance based on the binding ID included in the message. The binding between the binding ID and a service producer instance is stored within the Unit, e.g. framework function. The service producer instance may change, while the binding ID exposed to the Service Consumer remains the same. In that case different transactions may reach to different Service Instance even using the same binding ID.

The UE context are shared among the NF/Service Instance within the same NF/Service set. If one NF/Service Instance fails, the message targeting to the failed NF/Service Instance is routed to another NF/Service Instance within the same NF/Service set. The replacing NF/Service Instance retrieve the UE contexts to handle the incoming message. Thus the high reliability can be achieved. How to select the replacing NF/Service Instance is implemented as below:

- If not all the NF/Service Set instance within the same Units are failure, based on the binding ID type the Framework Function avoid select the failed NF/Service instance or reselect another NF Service Instance based on the preconfigured rule via OAM, e.g. when the NF/Service Instance-1 is failure it is replaced by NF/Service Instance -5.

- If a NF/Service Set is spread across multiple Units in different DCs, and all the NF/Service Set instance within the same Units are failure (which does not happen often) and if there are binding IDs bound to NF/Service instances in the failed Unit, the NRF notifies the communication peers of the NF/Service Set of the Unit failure event. Upon receiving the Unit failover notification, the communication peer re-selects a target Unit Address based on the NF/Service Set ID, and send messages to the target Unit using the same binding ID. The Service Framework Function in the target Unit selects a target NF/Service instance based on the binding ID and preconfigured rule.

- The replacing NF/Service Instance updates the Service Consumer instance with a new binding ID.

### 6.15.3 Illustrated procedures

The below procedure illustrate how to exchange the binding ID between the consumer and producer. And how the message is routed based on binding ID.



Figure 6.15.3-1 Binding information stored at the Framework Function and its usage

The binding between the service instance and the binding ID is maintained within the Unit, e.g. by the Framework Function. As an example, the binding can be established when the service instance is started, e.g. as part of the service instance registration procedure. The Service Framework includes a function module which stores the following information: the Service Set ID, Instance Pointer, IP address. Thus no matter which type binding ID is used by the service instance later, the Function in the Unit, e.g. Framework Function, can always route the message to the service instance. The service instance indicates the assigned binding ID to the Service Framework at the registration procedure, and Service Framework stores the assigned binding ID.

Binding ID exchange between the consumer and producer:

1. The consumer allocates a binding ID, which is related to the service consumer and used for following transaction request from the peer service instance, and include this information in the message sent to producer. The type of binding ID consumer allocated is per how the consumer prefer following transaction request from peer side communicate with it.

If the following transaction request from peer side is preferred to be handled by any instance within the same service consumer set, the binding ID is Service Set ID based. If the following transaction request from peer side is preferred to be handled by this instance, the binding ID is Service Set ID and Instance Pointer based.

NOTE: the consumer's binding ID is included only if the consumer can behave as service producer

2. The Function in the Unit, e.g. the framework function, selects the producer instance based on the previously stored association of binding ID and instance ID.

3. The Message 1 is forwarded to the selected producer instance.

4. The producer instance provides a producer's binding ID to the consumer instance in response message. The type of binding ID allocated is similar as the step 1.

5. The response message is forwarded to the Consumer. The Consumer stores the received Producer's binding ID as part of the UE context.

Binding ID usage for the following transaction:

6. Consumer sends message 2, including producer's binding ID received at step 5.

7. If the producer's binding ID allocated at step 4 is the Service Set ID and Instance pointer based, Producer 1 is selected based on producer's binding ID.

If the producer's binding ID allocated at step 4 is the Service Set ID based, the Function in the Unit, e.g. the framework function, re-selects the producer instance. The re-selected producer instance may be different comparing to the Producer 1. In that case if the transaction need be routed to the same Producer Instance for following transaction, another information need be provided, e.g. a different binding information which is called as temporary binding ID defined in clause 6.9 is used.

8. Message 2 is forwarded to Producer 1.

Binding information update:

9. The binding between the binding ID and Producer 1 is released, e.g. due to producer instance is deregistered scale in/out.

Message handling after the binding information is released:

10. The consumer sends Message 3 which include the producer's binding ID provided by Producer 1.

11. Since there is no producer instance associated with the binding ID, but the binding ID includes the Service Set ID information, a new producer instance is selected based on Producer service set ID and optional preconfigured rule.12. Message 3 is forwarded to Producer 2.

Editor's note: Producer 2 receives the request with the Pointer of Producer 1. It is FFS how the producer handles this mismatch.

13. The Producer 2 retrieve the UE context and provides a new producer's binding ID which is associated with producer 2 or this Set.

14. The response message is forwarded to the Consumer.

### 6.15.4 Impacts on existing NFs, NF Services and Interfaces

The impact of this solution includes:

- The Framework function manage the mapping between the producer NF/Service instance and binding ID. The Service Framework update the association when the producer instances are changed.

- The Producer NF/Service instance generate and send the binding ID to consumer NF/Service instance. The Consumer NF/Service instance store the received binding for the following transaction with producer NF/Service Instance. The Producer NF/Service instance may update the binding ID and send to the consumer NF/Service Instance anytime.

- The consumer NF/Service instance includes the binding ID when it communicate with the producer NF/Service instance. The Service Framework must be able to target the service request to the corresponding instance based on the binding ID in the service request.

- The Service Framework Function route messages based on binding ID to corresponding NF/Service instance.

### 6.15.5 Evaluation of the Solution

This solution focus on how to maintain the communication between the consumer and producer, i.e. via the binding ID. How to share the data among the NF/Service Instance within the same NF/Service Set is independent on how long the binding is maintained.

This solution have following characteristics:

- It provides a method to setup a long-living binding between the service consumer and producer. The binding can be set either with the service Set ID or Service Set ID and Instance Pointer. The binding is released when the UE context is released, e.g. then the PDU Session is released.

- The producer instances can decide how to bind a consumer instance with a producer instance, for example, a consumer instance can be bound to a producer instance only for a transaction, or for a period of time (e.g. it is maintained until the producer set is rescaled).

- The Service Framework Function maintains the binding between the binding ID and the producer NF/Service instance. The Service Framework can update the binding to a new producer instance in middle of the PDU Session e.g. when the producer instances are deregistered. The consumer NF/Service instance does not need to be aware which producer NF/Service instance is bound to the binding ID.

* Support partial or all the NF/Service Instance within one Unit are failure.

By using the binding ID it avoid the unnecessity to change the NF Service Instance for each transaction but also keep the flexibility if the NF/Service Instance change per transaction is required.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* End of Changes \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/