NRF功能说明

NRF（network function repository function）把NF Instance（网络资源）以profile的形式存储起来，并可被其它NF Instance发现和利用

NRF的功能主要分为三部分：NFManagement， NFDiscovery和Assess Token（OAuth2 Authorization）

协议文档29510为描述NRF NFManagement和NFDiscovery的主要文档

NRF运行起来之后情况如下：

```
    Finished dev [unoptimized + debuginfo] target(s) in 0.29s
     Running `target/debug/sim-nrf`
Started sim-nrf: 0.0.0.0:80 with DB at localhost:6379
```

终端窗口将会停在这里，其功能的测试需要另外开一个窗口进行命令操作，从代码上看，NRF本质是一个http服务器，测试NRF可以使用curl发送http包，也可以用rust test和pytest进行测试

一.NFManagement

NFManagement的功能分为：NFRegister，NFUpdate ， NFDeregister， NFStatusSubscribe， NF Heart-beat，NFStatusUnsubscribe， NFStatusNotify， NFListRetrieval， NFProfileRetrieval

1.NFRegister

向NRF注册一个新的NF Instance，注册时需要提供profile用以描述所注册的NF Instance，profile为json的形式，成功返回201，并返回uri和heart-beat时间，由于json格式错误导致失败则返回400，由于NRF内部错误则返回500，可支持的注册NF Instance类型包括

Table 6.1.6.3.3-1: Enumeration NFType

Enumeration value	Description
"NRF"	Network Function: NRF
"UDM"	Network Function: UDM
"AMF"	Network Function: AMF
"SMF"	Network Function: SMF
"AUSF"	Network Function: AUSF
"NEF"	Network Function: NEF
"PCF"	Network Function: PCF
"SMSF"	Network Function: SMSF
"NSSF"	Network Function: NSSF
"UDR"	Network Function: UDR
"LMF"	Network Function: LMF
"GMLC"	Network Function: GMLC
"5G_EIR"	Network Function: 5G-EIR
"SEPP"	Network Function: SEPP
"UPF"	Network Function: UPF
"N3IWF"	Network Function: N3IWF
"AF"	Network Function: AF
"UDSF"	Network Function: UDSF
"BSF"	Network Function: BSF
"CHF"	Network Function: CHF
"NWDAF"	Network Function: NWDAF

也支持用户自己定义的类型，不过需要加上“CUSTOM_”作为前缀

可用curl命令测试：

curl --verbose -d '{"nfInstanceId":"smf1-smf2", "nfType":"SMF", "nfStatus":"SUSPENDED", "fqdn":"casa.host12345.com", "ipv4Addresses": ["172.17.1.2"], "ipv6Addresses": ["2001:db8:abcd:0012::1"], "ipv6Prefixes": ["2001:db8::/32"], "nfServices":[{"serviceInstanceId":"id", "serviceName":"smf1", "version":[{"apiVersionInUri":"uri","apiFullVersion":"version1"}], "schema":"schema"}]}' -H "Content-Type: application/json" -X PUT http://localhost:80/nnrf-nfm/v1/nf-instances/smf1-smf2

其中-d代表用POST方法发送数据，NFregister时后面接的内容为NF profile，发送的数据都为json格式

注册成功情况如下：

```
root@wlan16:~# curl --verbose -d '{"nfInstanceId":"smf1-smf2", "nfType":"SMF", "nfStatus":"SUSPENDED", "fqdn":"casa
.host12345.com", "ipv4Addresses": ["172.17.1.2"], "ipv6Addresses": ["2001:db8:abcd:0012::1"], "ipv6Prefixes": ["200
1:db8::/32"], "nfServices":[{"serviceInstanceId":"id", "serviceName":"smf1", "version":[{"apiVersionInUri":"uri","a
piFullVersion":"version1"}], "schema":"schema"}]}' -H "Content-Type: application/json" -X PUT http://localhost:80/n
nrf-nfm/v1/nf-instances/smf1-smf2
*   Trying ::1...
* TCP_NODELAY set
* connect to ::1 port 80 failed: Connection refused
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 80 (#0)
> PUT /nnrf-nfm/v1/nf-instances/smf1-smf2 HTTP/1.1
> Host: localhost
> User-Agent: curl/7.58.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 360
>
* upload completely sent off: 360 out of 360 bytes
< HTTP/1.1 201 Created
< content-length: 645
< content-type: application/json
< date: Thu, 14 Feb 2019 01:55:05 GMT
<
* Connection #0 to host localhost left intact
{"heartBeatTimer":300,"nfProfile":{"nfInstanceId":"smf1-smf2","nfType":"SMF","nfStatus":"SUSPENDED","fqdn":"casa.ho
st12345.com","ipv4Addresses":["172.17.1.2"],"ipv6Addresses":["2001:db8:abcd:0012::1"],"ipv6Prefixes":["2001:db8::/3
2"],"nfServices":[{"serviceInstanceId":"id","serviceName":"smf1","version":[{"apiVersionInUri":"uri","apiFullVersio
n":"version1","expiry":null}],"schema":"schema","fqdn":null,"interPlmnFqdn":null,"ipEndPoints":null,"apiPrefix":nul
l,"defaultNotificationSubscriptions":null,"allowedPlmns":null,"allowedNfTypes":null,"allowedNfDomains":null,"allowe
dNssais":null,"capacity":null,"load":null,"supportedFeatures":null}]}}root@wlan16:~#
```

NRF打印如下：

```
NRFmanagement put nf-instances/{nfInstanceID}
HttpRequest HTTP/1.1 PUT:/nnrf-nfm/v1/nf-instances/smf1-smf2
 params: Params { url: Url { uri: /nnrf-nfm/v1/nf-instances/smf1-smf2, path: None }, tail: 35, segments:
[("nfInstanceID", UrlSegment(26, 35))] }
 headers:
 "host": "localhost"
 "user-agent": "curl/7.58.0"
 "accept": "*/*"
 "content-type": "application/json"
 "content-length": "360"

Return : ["Created"]
set/reset exiration timer with key "NRFTimer:NRFProfile:Id-NfProfile:smf1-smf2" to 300 seconds
```

NRF打印返回的为Created， 如果该NF Instance之前已经注册过（该NF Instance的ID存在于数据库），那将认为是NFUpdate，会将之前的profile整个替换掉，并返回200，打印的值为Replaced

2.NFUpdate

向NRF更新之前注册的NF Instance profile，可以使用PUT方法更新整个profile，也可以使用PATCH方法更新profile的部分参数，PATCH的操作要支持add/delete/replace，成功返回200，由于json格式错误导致失败则返回400，由于NRF内部错误则返回500，在NRF中找不到NF instance的ID则返回404

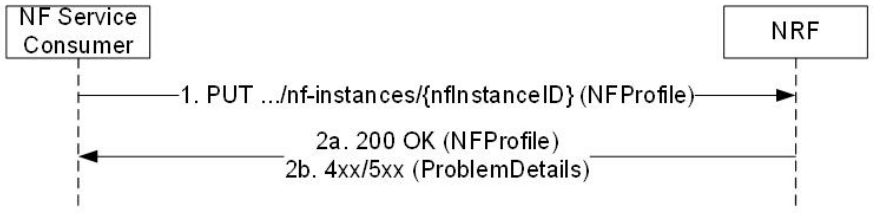
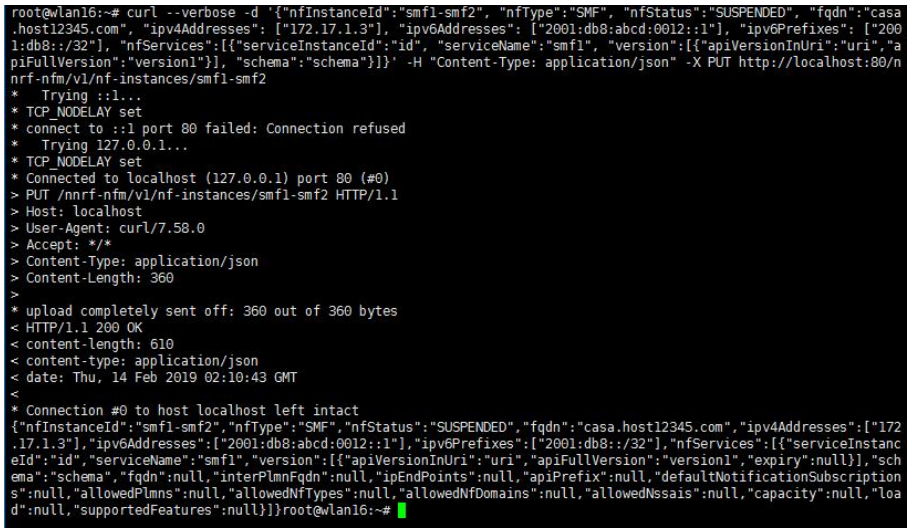


Figure 5.2.2.3.1-1: NF Profile Complete Replacement

curl命令测试如下，整个替换的话跟NFRegister差不多，比如把上面注册的那个NF Instance的ipv4Addresses改为172.17.1.3：

curl --verbose -d '{"nfInstanceId":"smf1-smf2", "nfType":"SMF", "nfStatus":"SUSPENDED", "fqdn":"casa.host12345.com", "ipv4Addresses": ["172.17.1.3"], "ipv6Addresses": ["2001:db8:abcd:0012::1"], "ipv6Prefixes": ["2001:db8::/32"], "nfServices":[{"serviceInstanceId":"id", "serviceName":"smf1", "version":[{"apiVersionInUri":"uri","apiFullVersion":"version1"}], "schema":"schema"}]}' -H "Content-Type: application/json" -X PUT http://localhost:80/nnrf-nfm/v1/nf-instances/smf1-smf2

成功返回200和修改后的NF profile：



打印则返回的是Replaced：

```
NRFmanagement put nf-instances/{nfInstanceID}
HttpRequest HTTP/1.1 PUT:/nnrf-nfm/v1/nf-instances/smf1-smf2
 params: Params { url: Url { uri: /nnrf-nfm/v1/nf-instances/smf1-smf2, path: None }, tail: 35, segments: [("nfInstanceID", UrlSegment(26, 35))] }
 headers:
 "host": "localhost"
 "user-agent": "curl/7.58.0"
 "accept": "*/*"
 "content-type": "application/json"
 "content-length": "360"

Return ["Replaced"]
set/reset exiration timer with key "NRFTimer:NRFProfile:Id-NfProfile:smf1-smf2" to 300 seconds
```

或者使用PATCH命令对已经注册过的NF Instance的profile进行部分更新，首先注册一个NF Instance

curl --verbose -d '{"nfInstanceId":"10","nfType":"SMF","nfStatus":"REGISTERED"}' -H "Content-Type: application/json" -X PUT http://localhost:80/nnrf-nfm/v1/nf-instances/10

这个NF Instance的profile是不完整的，通过PATCH可以对它进行add，copy，move，remove，replace和test操作

add：

curl --verbose -d '[{"op": "add", "path": "/nfServices", "value":[]}, {"op": "add", "path": "/nfServices/0", "value": {}}, {"op": "add", "path": "/nfServices/0/serviceInstanceId", "value": "100"},{"op": "add", "path": "/nfServices/0/serviceName", "value": "servicename-1"},{"op": "add", "path": "/nfServices/0/version", "value": [{"apiVersionInUri":"uri-1", "apiFullVersion":"version-1"}]},{"op": "add", "path": "/nfServices/0/schema", "value": "schema-1"}]' -H "Content-Type: application/json" -X PATCH http://localhost:80/nnrf-nfm/v1/nf-instances/10

可通过add操作补充profile json中的内容

copy:

curl --verbose -d '[{"op": "copy", "path": "/nfServices/1", "from": "/nfServices/0"}]' -H "Content-Type: application/json" -X PATCH http://localhost:80/nnrf-nfm/v1/nf-instances/10

这个copy操作是把profile json中的nfServices的第一个成员复制一个作为nfServices的第二个成员

move:

curl --verbose -d '[{"op": "add", "path": "/capacity", "value": 99}, {"op": "move", "path": "/nfServices/1/capacity", "from": "/capacity"}]' -H "Content-Type: application/json" -X PATCH http://localhost:80/nnrf-nfm/v1/nf-instances/10

这个操作是把profile json中的capacity移动到nfServices的第一个成员下

remove:
curl --verbose -d '[{"op": "remove", "path": "/nfServices/1/capacity"}]' -H "Content-Type: application/json" -X PATCH http://localhost:80/nnrf-nfm/v1/nf-instances/10
移除profile json中的nfServices的第一个成员的capacity

replace:
curl --verbose -d '[{"op": "replace", "path": "/nfServices/1/serviceInstanceId", "value": "200"}, {"op": "replace", "path": "/nfServices/1/serviceName", "value": "servicename-2"}, {"op": "replace", "path": "/nfServices/1/version", "value": [{"apiVersionInUri":"uri-2", "apiFullVersion":"version-2"}]}, {"op": "replace", "path": "/nfServices/1/schema", "value": "schema-2"}]' -H "Content-Type: application/json" -X PATCH http://localhost:80/nnrf-nfm/v1/nf-instances/10

test:
curl --verbose -d '[{"op": "test", "path": "/nfServices/1/version/0/apiVersionInUri", "value": "uri-2"}]' -H "Content-Type: application/json" -X PATCH http://localhost:80/nnrf-nfm/v1/nf-instances/10
****
PATCH的时候必须严格符合数据格式和路径，否则会导致panic， 比如add时指定的路径是profile的json格式中没有的成员，就会导致panic

3.NFDeregister
注销之前注册在NRF的NF instance，NRF会删除其profile，成功返回204，失败返回404代表其找不到对应的NF instance ID

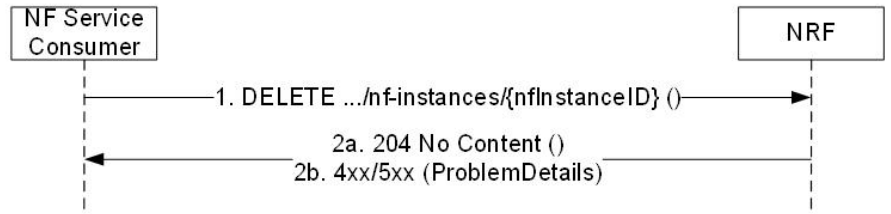


Figure 5.2.2.4.1-1: NF Instance Deregistration

curl命令如下：
curl --verbose -H "Content-Type: application/json" -X DELETE http://localhost:80/nnrf-nfm/v1/nf-instances/smf1-smf2

```
root@wlan16:~# curl --verbose -H "Content-Type: application/json" -X DELETE http://localhost:80/nnrf-nfm/v1/nf-instances/smf1-smf2
*   Trying ::1...
* TCP_NODELAY set
* connect to ::1 port 80 failed: Connection refused
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 80 (#0)
> DELETE /nnrf-nfm/v1/nf-instances/smf1-smf2 HTTP/1.1
> Host: localhost
> User-Agent: curl/7.58.0
> Accept: */*
> Content-Type: application/json
>
< HTTP/1.1 204 No Content
< content-length: 0
< date: Thu, 14 Feb 2019 02:56:10 GMT
<
* Connection #0 to host localhost left intact
```

4.NFStatusSubscribe
Consumer订阅关注已经注册在NRF上的某些NF instance

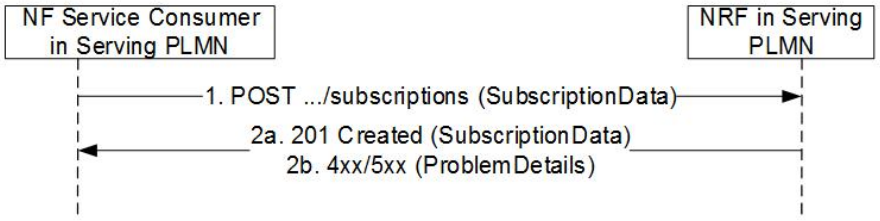


Figure 5.2.2.5.2-1: Subscription to NF Instances in the same PLMN

用post方法向NRF发送一个subscriptionData，这个subscriptionData包括一些“特定的条件”和一个用于接受宣告信息（notification）的uri，“特定的条件”用于NRF筛选出Consumer感兴趣NF instance，“特定的条件”同时也描述了NRF什么时候需要向Consumer发送宣告信息，如下所示：

Table 6.1.6.2.16-1: Definition of type SubscriptionData

Attribute name	Data type	P	Cardinality	Description
nfStatusNotificationUri	Uri	M	1	Callback URI where the NF Service Consumer will receive the notifications from NRF.
subscrCond	SubscrCond	O	0..1	If present, this attributed shall contain the conditions identifyting the set of NF Instances whose status is requested to be monitored.If this attribute is not present, it means that the NF Service Consumer requests a subscription to all NFs in the NRF (NOTE).
subscriptionId	string	C	0..1	Subscription ID for the newly created resource. This parameter shall be absent in the request to the NRF and shall be included by NRF in the response to the subscription creation request. Read-Only: true
validityTime	DateTime	C	0..1	Time instant after which the subscription becomes invalid. This parameter may be sent by the client, as a hint to the server, but it shall be always sent back by the server (regardless of the presence of the attribute in the request) in the response to the subscription creation request.
reqNotifEvents	array(NotificationEventType)	O	1..N	If present, this attribute shall contain the list of event types that the NF Service Consumer is interested in receiving. If this attribute is not present, it means that notifications for all event types are requested.
reqNfType	NFType	O	0..1	If included, this IE shall contain the NF type of the NF Service Consumer that is requesting the creation of the subscription. The NRF shall use it for authorizing the request, in the same way as the "requester-nf-type" is used in the NF Discovery service (see Table 6.2.3.2.3.1-1).
reqNfFqdn	Fqdn	O	0..1	If included, this IE shall contain the FQDN of the NF Service Consumer that is requesting the creation of the subscription. The NRF shall use it for authorizing the request, in the same way as the "requester-nf-instance-fqdn" is used in the NF Discovery service (see Table 6.2.3.2.3.1-1).
plmnId	PlmnId	O	0..1	If present, this attribute contains the target PLMN ID of the NF Instance(s) whose status is requested to be monitored.
notifCondition	NotifCondition	O	0..1	If present, this attribute contains the conditions that trigger a notification from NRF; this attribute shall only be present if the NF Service Consumer has subscribed to changes on the NF Profile (i.e., reqNotifEvents contains the value "NF_PROFILE_CHANGED", or reqNotifEvents attribute is absent). If this attribute is absent, it means that the NF Service Consumer does not indicate any restriction, or condition, on which attributes of the NF Profile shall trigger a notification from NRF.
NOTE: The "subscription to all NFs" may be quite demanding in terms of resources in NRF and also in terms of network traffic of the resulting notifications, so it should be authorized by NRF under very strict policies (e.g. only to a specific requesting NF, as indicated by reqNfType and reqNfFqdn attributes).				

注册成功返回201，同时返回一个NRF分配的subsciption ID（在29510-f20的版本还将返回有效时间，超过有效时间，Consumer将不能再收到宣告信息），由于json格式错误导致失败则返回400，由于NRF内部错误则返回500

curl命令如下：

curl --verbose -d '{"nfStatusNotificationUri": "http://test_nrf:42767/nrf-subscribe-notify/nfinstance-1"}' -H "Content-Type: application/json" -X POST http://localhost:80/nnrf-nfm/v1/subscriptions

其中nfStatusNotificationUri所给出的uri就是用来接收nrf发送出来的notification的，所以如果想验证nrf的NFStatusNotify功能，必须确保这个uri是真实有效的，subscribe成功则返回一个subscription ID，这个ID可用于NFStatusUnsubscribe：

```
root@wlan16:~# curl --verbose -d '{"nfStatusNotificationUri": "http://test_nrf:42767/nrf-subscribe-notify/nfinstanc
e-1"}' -H "Content-Type: application/json" -X POST http://localhost:80/nnrf-nfm/v1/subscriptions
Note: Unnecessary use of -X or --request, POST is already inferred.
*   Trying ::1...
* TCP_NODELAY set
* connect to ::1 port 80 failed: Connection refused
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 80 (#0)
> POST /nnrf-nfm/v1/subscriptions HTTP/1.1
> Host: localhost
> User-Agent: curl/7.58.0
> Accept: */*
> Content-Type: application/json
> Content-Length: 86
>
* upload completely sent off: 86 out of 86 bytes
< HTTP/1.1 201 Created
< content-length: 3
< content-type: application/json
< date: Thu, 14 Feb 2019 03:29:05 GMT
<
* Connection #0 to host localhost left intact
"1"root@wlan16:~#
root@wlan16:~#
root@wlan16:~#
```

****

我们的NRF目前是根据TS29510-f00做的，不会返回有效时间（validityTime），而且TS29510-f20所描述的以下的subscribe的功能我们的NRF上暂无

NRF支持Consumer订阅关注处于不同PLMN的NRF上的NF Instance

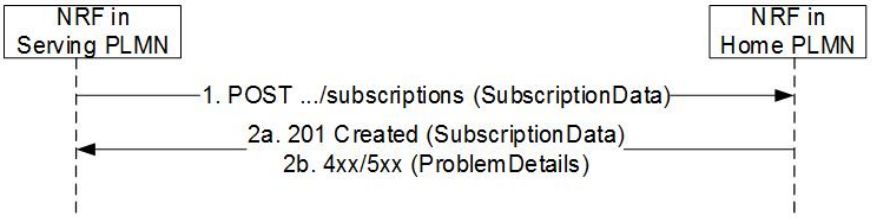


Figure 5.2.2.5.3-1: Subscription to NF Instances in a different PLMN

此时返回值需要加上MCC（移动国家码）和MNC（移动网络号码）

EXAMPLE: If the NRF in a Home PLMN (where MCC = 123, and MNC=456) creates a subscription with value "subs987654", the subscriptionID that the NRF in Serving PLMN would send to the NF Service Consumer in Serving PLMN is: "123456-subs987654"

Consumer也可以通过同一个PLMN的NRF订阅关注另一个NRF上的NF Instance

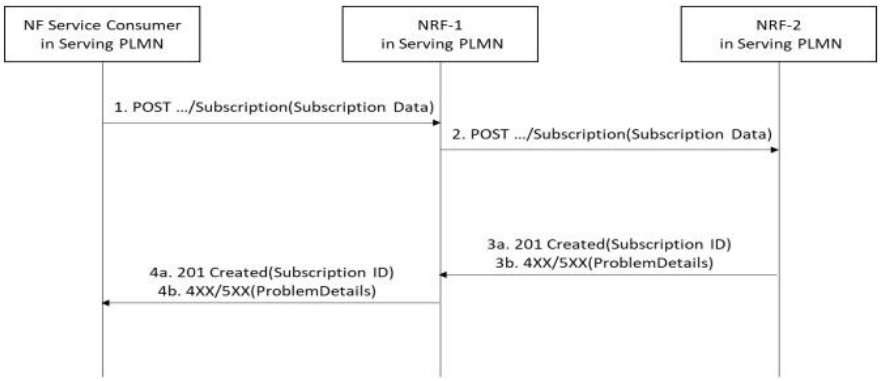


Figure 5.2.2.5.4-1: Subscription with intermediate forwarding NRF

NRF如何判断Consumer所要关注的NF Instance在另一个NRF上，是通过FQDN吗？

当有多个NRF时，NRF的subscription功能还支持重定向

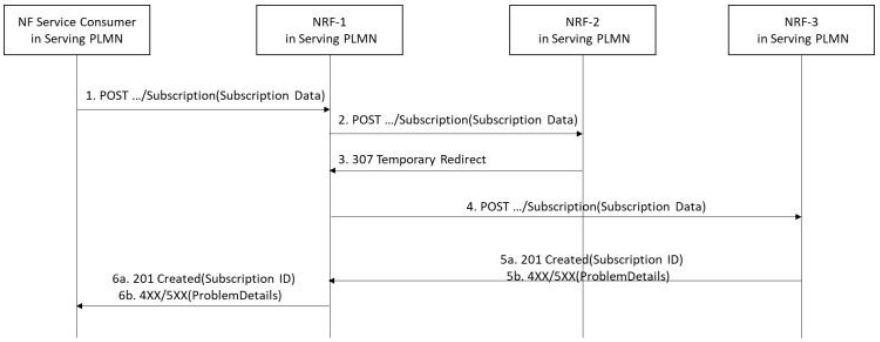


Figure 5.2.2.5.5-1: Subscription to NF Instances with intermediate redirecting NRF

subscription的重定向应该跟http头里的location有关，但具体流程不是很清楚

如果subscription超过有效时间，可用patch方法更新之前的NFStatusSubscribe，刷新有效时间（只能更新有效时间），如果NRF为这个subscription分配的有效时间跟request时提供的一致，返回204，不一样则返回200

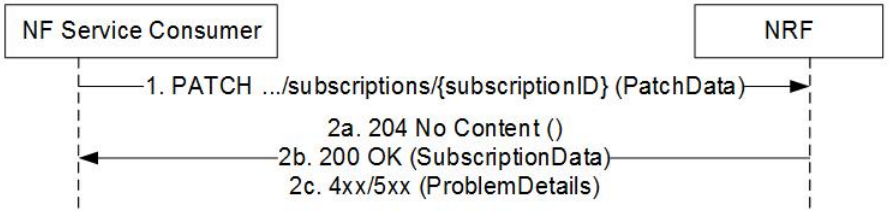


Figure 5.2.2.5.6-1: Subscription to NF Instances in the same PLMN

5.NF Heart-beat

当NF Instance注册成功时会返回一个心跳时间(heart-beat times)，NF Instance需要每隔一段时间用PATCH方法向NRF发送信息，告知自己的并无异常。这个时间要小于NRF的心跳时间，否则NRF认为该NF Instance发生异常，并将数据库中存储的对应的NF Instance的状态改为挂起（SUSPEND），此时，该NF Instance将不能再被Consumer用NFdiscovery发现和利用

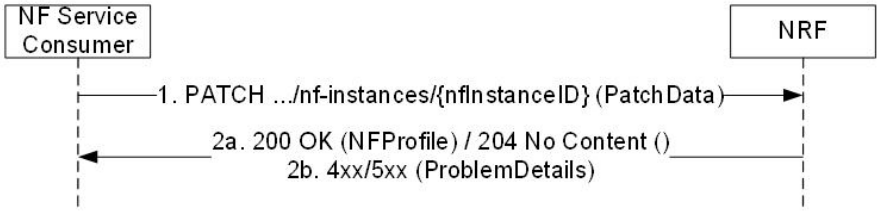


Figure 5.2.2.3.2-1: NF Heart-Beat

PATCH的时候operation应该用“replace”，并将profile里nfStatus的值设为“REGISTER”，如下所示

```
[
  { "op": "replace", "path": "/nfStatus", "value": "REGISTERED" },
  { "op": "replace", "path": "/load", "value": 50 }
]
```

****

这个功能目前NRF上只会在NFRegister成功时返回心跳时间，心跳时间默认为300s，并没有动态监测的功能

6.NFStatusNotify

当Consumer所订阅关注的NF Instance的状态发生改变时，NRF用POST方法向之前NFStatusSubscribe时所提供的URI发送notification

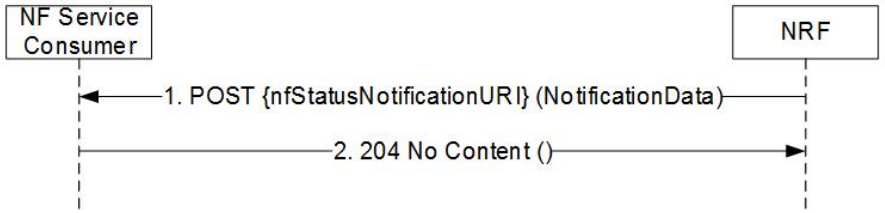


Figure 5.2.2.6.2-1: NRF Notifications

有四种情况NRF需要发送notification：①新注册了NF Instance ②已经注册的NF Instance的内容被Consumer用PATCH更新了③NF Instance被注销掉了④NRF收不到已经完成注册的NF Instance 的heart-beat，NRF将自动将NF Instance的状态改为挂起并告知对这个NF Instance进行subsribe的Consumer（④是TS29510上才有规定的）

notification的内容如下所示：

Table 6.1.6.2.17-1: Definition of type NotificationData

Attribute name	Data type	P	Cardinality	Description
event	NotificationEventType	M	1	Notification type. It shall take the values "NF_REGISTERED", "NF_DEREGISTERED" OR "NF_PROFILE_CHANGED".
nfInstanceUri	Uri	M	1	Uri of the NF Instance (see subclause 6.1.3.3.2) associated to the notification event.
nfProfile	NFProfile	C	0..1	New NF Profile or Updated NF Profile; it shall be present when the notification type is "NF_REGISTERED" or "NF_PROFILE_CHANGED".
profileChanges	array(ChangeItem)	C	1..N	List of changes on the profile of the NF Instance associated to the notification event; it may be present when the notification type is "NF_PROFILE_CHANGED" (see NOTE 1).
NOTE 1: If "event" attribute takes the value "NF_PROFILE_CHANGED", then either "nfProfile" or "profileChanges" attributes shall be present, but not both.				

EXAMPLE: Notification payload sent from NRF when an NF Instance has changed its profile by updating the value of the "recoveryTime" attribute of its NF Profile, and updated the TCP port of the first NF Service Instance:

```
{
  "event": "NF_PROFILE_CHANGED",
  "nfInstanceUri": "../nf-instances/4947a69a-f61b-4bc1-b9da-47c9c5d14b64",
  "profileChanges": [
    {
      "op": "REPLACE",
      "path": "/recoveryTime",
      "newValue": "2018-12-30T23:20:50Z"
    },
    {
      "op": "REPLACE",
      "path": "/nfServices/0/ipEndPoints/0/port",
      "newValue": 8080
    }
  ]
}
```

NFNotify的功能无法通过curl测试，测试需要用rust或者pytest搭建起一个http服务器用以接受NRF发出来的notification，并且在subscribe的时候要把这个http服务器的正确的uri发给NRF

7.NFStatusUnsubscribe

取消对之前NF Instance的订阅关注， 成功返回204，失败返回404代表其找不到对应的NF instance ID

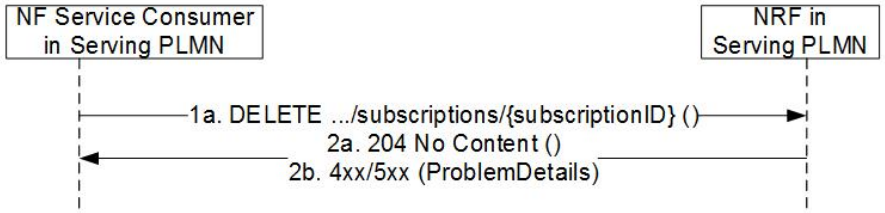


Figure 5.2.2.7.2-1: Subscription removal in the same PLMN

curl命令如下所示：

curl --verbose -d -H "Content-Type: application/json" -X DELETE http://localhost:80/nnrf-nfm/v1/subscriptions/1

其中，uri最后的“1”是subscribe成功时返回回来的

```
root@wlan16:~/sim-nrf/src/api# curl --verbose -d -H "Content-Type: application/json" -X DELETE http://localhost:80/
nnrf-nfm/v1/subscriptions/1
* Port number ended with ' '
* Closing connection -1
curl: (3) Port number ended with ' '
*   Trying ::1...
* TCP_NODELAY set
* connect to ::1 port 80 failed: Connection refused
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 80 (#0)
> DELETE /nnrf-nfm/v1/subscriptions/1 HTTP/1.1
> Host: localhost
> User-Agent: curl/7.58.0
> Accept: */*
> Content-Length: 2
> Content-Type: application/x-www-form-urlencoded
>
* upload completely sent off: 2 out of 2 bytes
< HTTP/1.1 204 No Content
< content-length: 0
< date: Thu, 14 Feb 2019 05:53:55 GMT
<
* Connection #0 to host localhost left intact
```

8.NFListRetrieval

检索当前NRF上注册的NF Instance，并返回一个URI列表

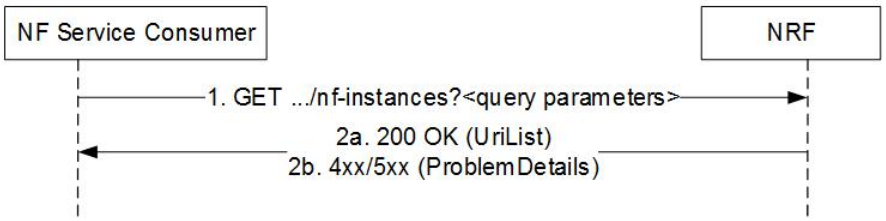


Figure 5.2.2.8.1-1: NF instance list retrieval

可设置query paramter ，可选的query paramter有nf-type和limit，nf-type对NF Instance进行类型筛选，limit则设置列表项的上限值，比如limit = 10则返回的列表的URI将不超过10个

curl命令如下所示：

curl --verbose -H "Content-Type: application/json" -X GET http://localhost:80/nnrf-nfm/v1/nf-instances?nf-type=NRF\&limit=10

在curl命令中“&”符号无法识别到，需要使用\&进行转义

成功则返回一个符合条件的uri列表：

```
root@wlan16:~/sim-nrf/tests# curl --verbose -H "Content-Type: application/json" -X GET http://172.18.0.1:80/nnrf-nfm/v1/nf-instances?nf-type=SMF\&limit=10
Note: Unnecessary use of -X or --request, GET is already inferred.
*   Trying 172.18.0.1...
* TCP_NODELAY set
* Connected to 172.18.0.1 (172.18.0.1) port 80 (#0)
> GET /nnrf-nfm/v1/nf-instances?nf-type=SMF&limit=10 HTTP/1.1
> Host: 172.18.0.1
> User-Agent: curl/7.58.0
> Accept: */*
> Content-Type: application/json
>
< HTTP/1.1 200 OK
< content-length: 115
< content-type: application/json
< date: Thu, 14 Feb 2019 06:53:58 GMT
<
* Connection #0 to host 172.18.0.1 left intact
["/nnrf-nfm/v1/nf-instances/smf1-smf2","/nnrf-nfm/v1/nf-instances/smf1-smf4","/nnrf-nfm/v1/nf-instances/smf1-smf3"]
```

9.NFProfileRetrieval

进行NFProfileRetrieval时需要给定一个NF Instance ID，NRF检索当前NRF上已经注册的全部NF Instance，并把检索到的NF Instance的profile返回回来，成功返回200，找不到ID对应的NF Instance返回404，错误返回403或者500

Figure 5.2.2.9.1-1: NF profile retrieval

curl命令测试如下：

curl --verbose -H "Content-Type: application/json" GET http://localhost:80/nnrf-nfm/v1/nf-instances/smf1-smf2

执行结果如下所示：

```
root@wlan16:~/sim-nrf/src# curl --verbose -H "Content-Type: application/json" GET http://localhost:80/nnrf-nfm/v1/nf-instances/smf1-smf2
* Rebuilt URL to: GET/
* Could not resolve host: GET
* Closing connection 0
curl: (6) Could not resolve host: GET
*   Trying ::1...
* TCP_NODELAY set
* connect to ::1 port 80 failed: Connection refused
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 80 (#1)
> GET /nnrf-nfm/v1/nf-instances/smf1-smf2 HTTP/1.1
> Host: localhost
> User-Agent: curl/7.58.0
> Accept: */*
> Content-Type: application/json
>
< HTTP/1.1 200 OK
< content-length: 610
< content-type: application/json
< date: Fri, 15 Feb 2019 09:51:03 GMT
<
* Connection #1 to host localhost left intact
{"nfInstanceId":"smf1-smf2","nfType":"SMF","nfStatus":"SUSPENDED","fqdn":"casa.host12345.com","ipv4Addresses":["172.17.1.2"],"ipv6Addresses":["2001:db8:abcd:0012::1"],"ipv6Prefixes":["2001:db8::/32"],"nfServices":[{"serviceInstanceId":"id","serviceName":"smf1","version":[{"apiVersionInUri":"uri","apiFullVersion":"version1","expiry":null}],"schema":"schema","fqdn":null,"interPlmnFqdn":null,"ipEndPoints":null,"apiPrefix":null,"defaultNotificationSubscriptions":null,"allowedPlmns":null,"allowedNfTypes":null,"allowedNfDomains":null,"allowedNssais":null,"capacity":null,"load":null,"supportedFeatures":null}]}root@wlan16:~/sim-nrf/src#
```

二.NFDiscovery

NFDiscovery只有一个功能NFDiscover，用于Consumer在NRF中查找网络资源

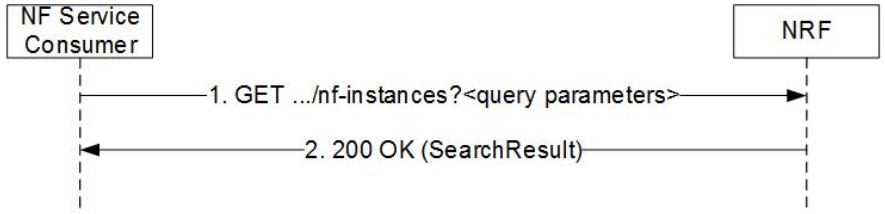


Figure 5.3.2.2.1-1: NF Discovery Request

查询时在query parameter中加入所需的条件，筛选的条件详细可以看TS29510的Table 6.2.3.2.3.1-1，其中service-names， target-nf-type和requester-nf-type在我们公司的NRF中这三个query parameter是必须要有的，不然会返回403

当查询成功时，NRF返回200，并把查询结果的有效周期和筛选出来的符合条件的NF Instance的profile返回回来

curl命令测试如下：

curl --verbose -H "Content-Type: application/json" -X GET http://localhost:80/nnrf-disc/v1/nf-instances?service-names=[%22smf%22]\&target-nf-type=SMF\&requester-nf-type=SMF

这里的%22是curl里面的 “” 的转义码，curl无法识别到某些转义字符，转义码可参考以下链接

https://blog.csdn.net/p312011150/article/details/78928003

运行结果如下：

```
root@wlan16:~/sim-nrf# curl --verbose -H "Content-Type: application/json" -X GET http://localhost:80/nnrf-disc/v1/nf-instances?service-names=[%22smf1%22]\&target-nf-type=SMF\&requester-nf-type=SMF
Note: Unnecessary use of -X or --request, GET is already inferred.
*   Trying ::1...
* TCP_NODELAY set
* connect to ::1 port 80 failed: Connection refused
*   Trying 127.0.0.1...
* TCP_NODELAY set
* Connected to localhost (127.0.0.1) port 80 (#0)
> GET /nnrf-disc/v1/nf-instances?service-names=[%22smf1%22]&target-nf-type=SMF&requester-nf-type=SMF HTTP/1.1
> Host: localhost
> User-Agent: curl/7.58.0
> Accept: */*
> Content-Type: application/json
>
< HTTP/1.1 200 OK
< content-length: 2188
< content-type: application/json
< date: Mon, 18 Feb 2019 05:40:33 GMT
<
{"validityPeriod":30,"nfInstances":[{"nfInstanceId":"smf1-smf2","nfType":"SMF","plmn":null,"sNssais":null,"nsiList":null,"fqdn":"casa.host12345.com","ipv4Addresses":["172.17.1.2"],"ipv6Addresses":["2001:db8:abcd:0012::1"],"ipv6Prefixes":["2001:db8::/32"],"capacity":null,"load":null,"locality":null,"priority":100,"udrInfo":null,"udmInfo":null,"ausfInfo":null,"amfInfo":null,"smfInfo":null,"upfInfo":null,"pcfInfo":null,"bsfInfo":null,"nfServices":[{"serviceInstanceId":"id","serviceName":"smf1","version":[{"apiVersionInUri":"uri","apiFullVersion":"version1","expiry":null}],"schema":"schema","fqdn":null,"ipEndPoints":null,"apiPrefix":null,"defaultNotificationSubscriptions":null,"capacity":null,"load":null,"priority":100,"supportedFeatures":null}]},{"nfInstanceId":"smf1-smf4","nfType":"SMF","plmn":null,"sNssais":null,"nsiList":null,"fqdn":"casa.host12345.com","ipv4Addresses":["172.17.1.2"],"ipv6Addresses":["2001:db8:abcd:0012::1"],"ipv6Prefixes":["2001:db8::/32"],"capacity":null,"load":null,"locality":null,"priority":100,"udrInfo":null,"udmInfo":null,"ausfInfo":null,"amfInfo":null,"smfInfo":null,"upfInfo":null,"pcfInfo":null,"bsfInfo":null,"nfServices":[{"serviceInstanceId":"id","serviceName":"smf1","version":[{"apiVersionInUri":"uri","apiFullVersion":"version1","expiry":null}],"schema":"schema","fqdn":null,"ipEndPoints":null,"apiPrefix":null,"defaultNotificationSubscriptions":null,"capacity":null,"load":null,"priority":100,"supportedFeatures":null}]},{"nfInstanceId":"smf1-smf3","nfType":"SMF","plmn":null,"sNssais":null,"nsiList":null,"fqdn":"casa.host12345.com","ipv4Addresses":["172.17.1.2"],"ipv6Addresses":["2001:db8:abcd:0012::1"],"ipv6Prefixes":["2001:db8::/32"],"capacity":null,"load":null,"locality":null,"priority":100,"udrInfo":null,"udmInfo":null,"ausfInfo":null,"amfInfo":null,"smfInfo":null,"upfInfo":null,"pcfInfo":null,"bsfInfo":null,"nfServices":[{"serviceInstanceId":"id","serviceName":"smf1","version":[{"apiVersionInUri":"uri","apiFullVersion":"version1","expiry":null}],"schema":"schema","fqdn":null,"ipEndPo* Connection #0 to host localhost left intact
ints":null,"apiPrefix":null,"defaultNotificationSubscriptions":null,"capacity":null,"load":null,"priority":100,"supportedFeatures":null}]}]}root@wlan16:~/sim-nrf#
root@wlan16:~/sim-nrf#
```

目前有效周期validity period在代码中定死为30

三.Oauth2功能目前尚不完善

四.其他测试

pytest的测试首先要安装pip3

```
$ apt-get install python3-pip
```

如果下不了需要更换apt-get的源

再用pip3安装pytest和aiohttp

```
$ pip3 install pytest
$ pip3 install aiohttp
```

运行测试脚本

```
$ pytest xxx.py
```

rust test运行方法

```
$ cargo test test_nf_management_post_subscriptions_subscription_id -- --nocapture
```

直接运行cargo test会执行全部的#[test]下的函数，或者cargo test后面跟函数名，指定测试哪个函数

也可用--test指定测试某个文件，比如存在一个测试文件

1_PreR15_0_0_NRFNFManagementService_NFInstance_GET_URI.rs

则可以用如下方法运行

```
cargo test --test 1_PreR15_0_0_NRFNFManagementService_NFInstance_GET_URI
```

加了-- --nocapture这个选项后才会把#[test]下的函数里的打印输出到屏幕，不然将看不到rust test中的打印