Optimising IP subscriber configuration for a BNG deployment

Mar-2014

Roy Jiang (zhjiang@cisco.com)

Version 1.0
Executive summary

This document is intended to describe an approach to optimise the configuration for IP subscriber supported by cisco ASR9000 BNG feature set. We start the journey from a basic/original configuration for a typical BNG IP subscriber deployment scenario (DHCP triggered IPv4 subscriber with ASR9K imbedded DHCPv4 server), which works but not so optimized. We study what's the drawbacks, how to overcome them, and give optimised configurations from different perspective and dimension, analysing the benefits and side-effects of those optimisation.

To focus on our goal, other aspects of BNG configuration, such as AAA, QoS, control-policy, is not covered in this document. This document is based on the lab testing result using IOS XR 5.1.1 release image.

Deployment brief and original configuration

In this case, there are two DHCPv4 clients, named Apple (a MAC mini running MAC OS) and Mercury (an wireless AP) respectively. They come from the same access-intf, with same VLAN encapsulation.

The network between the DHCP clients and the BNG are pure L2 network. ASR9K here works as a DHCPv4 server.

Following is a brief of the topology.

There is no authentication, authorisation and accounting communication with an external RADIUS server.
Optimising IP subscriber configuration

An special thing in this deployment scenario is the using of multiple IPv4 subnets for addressing. To test the multiple address ranges in the same pool, here I have two subnets or networks configured, with default-router configured along with the each of the network. To simulate the case that DHCP server assigns IP from the second network when the first network is exhausted, we exclude all of the addresses but one for each of the network for testing purpose only, given I have only two DHCP clients here.

The int loopback 200 must have primary and secondary address, matching the two default-router, otherwise the ARP for the second default-router will get no reply from the BNG.

Following is the original configuration.
Optimising IP subscriber configuration

Original Config (not a complete one)

```
dhcp ipv4
profile SERVER_BASE server
lease 0 0 10
pool IPoE_POOL_BASE
dns-server 5.5.5.5 6.6.6.6

!interface Bundle-Ether1.1001 server profile SERVER_BASE
!
!pool vrf default ipv4 IPoE_POOL_BASE
network 20.0.0.0/24 default-router 20.0.0.1
network 21.0.0.0/24 default-router 21.0.0.1
exclude 20.0.0.0 20.0.0.253
exclude 21.0.0.0 21.0.0.253
!
interface Bundle-Ether1.1001
ipv4 point-to-point
ipv4 unnumbered Loopback200
service-policy type control subscriber CP_IPOE_DHCPV4_BASE
encapsulation dot1q 1 second-dot1q 1001
ipsubscriber ipv4 l2-connected
initiator dhcp
!
!
policy-map type control subscriber CP_IPOE_DHCPV4_BASE
event session-start match-first
class type control subscriber DHCPv4 do-until-failure
 10 activate dynamic-template DTP_IPOE_DHCPV4
!
!
end-policy-map
!

dynamic-template
type ipsubscriber DTP_IPOE_DHCPV4_BASE
ipv4 unnumbered Loopback200
!
!
interface Loopback200
description loopback for DHCP_triggered IP service
ipv4 address 20.0.0.1 255.255.255.0
ipv4 address 21.0.0.1 255.255.255.0 secondary
```

With above configuration, two CPEs get their sessions established with the addressing information summarised as below.
Optimising IP subscriber configuration

- DHCP Server identifier is the address that a DHCP client uses as destination IP for unicast based DHCP renew and release.
- Router is the Default Gateway assigned to the DHCP client, another word, when client send traffic out, it need firstly to send ARP request for the default GW address.

Following are the show CLI displays with original configuration for reference.

```
RP/0/RSP0/CPU0:Roy_BNG_1#sh dhcp ipv4 ser binding
Tue Feb 11 22:26:19.341 UTC

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>IP Address</th>
<th>State</th>
<th>Remaining</th>
<th>Interface</th>
<th>VRF</th>
<th>Sublabel</th>
</tr>
</thead>
<tbody>
<tr>
<td>3c07.545f.c041</td>
<td>20.0.0.254</td>
<td>BOUND</td>
<td>418</td>
<td>BE1.1001</td>
<td>default</td>
<td>0x53</td>
</tr>
<tr>
<td>a857.4e06.4f47</td>
<td>21.0.0.254</td>
<td>BOUND</td>
<td>340</td>
<td>BE1.1001</td>
<td>default</td>
<td>0xc9</td>
</tr>
</tbody>
</table>

RP/0/RSP0/CPU0:Roy_BNG_1#sh subscriber sess all
Tue Feb 11 22:26:40.084 UTC

Codes: IN - Initialize, CN - Connecting, CD - Connected, AC - Activated, ID - Idle, DN - Disconnecting, ED - End

<table>
<thead>
<tr>
<th>Type</th>
<th>Interface</th>
<th>State</th>
<th>Subscriber IP Addr / Prefix</th>
</tr>
</thead>
<tbody>
<tr>
<td>LNS Address</td>
<td>BE1.1001.ip1</td>
<td>AC</td>
<td>20.0.0.254 (default)</td>
</tr>
<tr>
<td></td>
<td>BE1.1001.ip2</td>
<td>AC</td>
<td>21.0.0.254 (default)</td>
</tr>
</tbody>
</table>

RP/0/RSP0/CPU0:Roy_BNG_1#sh route subscriber
Tue Feb 11 22:26:48.683 UTC

A 20.0.0.254/32 is directly connected, 01:07:34, Bundle-Ether1.1001.ip1
A 21.0.0.254/32 is directly connected, 01:06:19, Bundle-Ether1.1001.ip2

RP/0/RSP0/CPU0:Roy_BNG_1#sh arp
```
Optimising IP subscriber configuration

Wed Feb 12 00:29:31.051 UTC

-------------------------------------------------------------------------------

0/0/CPU0

-------------------------------------------------------------------------------

<table>
<thead>
<tr>
<th>Address</th>
<th>Age</th>
<th>Hardware Addr</th>
<th>State</th>
<th>Type</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>20.0.0.1</td>
<td>-</td>
<td>e4c7.223a.d1f3</td>
<td>Interface</td>
<td>ARPA</td>
<td>Bundle-Ether1.1001</td>
</tr>
<tr>
<td>20.0.0.1</td>
<td>-</td>
<td>e4c7.223a.d1f3</td>
<td>Interface</td>
<td>ARPA</td>
<td>Bundle-Ether1.1001.ip1</td>
</tr>
<tr>
<td>20.0.0.1</td>
<td>-</td>
<td>e4c7.223a.d1f3</td>
<td>Interface</td>
<td>ARPA</td>
<td>Bundle-Ether1.1001.ip2</td>
</tr>
<tr>
<td>21.0.0.1</td>
<td>-</td>
<td>e4c7.223a.d1f3</td>
<td>Interface</td>
<td>ARPA</td>
<td>Bundle-Ether1.1001</td>
</tr>
<tr>
<td>21.0.0.1</td>
<td>-</td>
<td>e4c7.223a.d1f3</td>
<td>Interface</td>
<td>ARPA</td>
<td>Bundle-Ether1.1001.ip1</td>
</tr>
<tr>
<td>21.0.0.1</td>
<td>-</td>
<td>e4c7.223a.d1f3</td>
<td>Interface</td>
<td>ARPA</td>
<td>Bundle-Ether1.1001.ip2</td>
</tr>
</tbody>
</table>
Optimising IP subscriber configuration for a BNG deployment

The focus on this document is to explain why the optimisation is needed and how to optimize. We should have the real deployment conduction in mind and I would like to optimize the configuration from following perspectives.

- Smoothness
- Simplicity
- Security
- Stability
- Scalability

**Optimization-1  Removing secondary IP**

Removing the secondary IP address from the unnumbered interface – a better way to support multiple subnets.

For sake of Smoothness and Simplicity and Scalability.

When you have to add another subnet to the address pool, it should be guaranteed that the existing services/sessions are not interrupted.

Let’s check the original configuration. On day one, I have only one subnet (20.0.0.0/24) configured, and some day later this subnet is exhausted, I need to add the second subnet, say 21.0.0.0/24. What I need to do is as following. The red CLI on the right is the CLI I need to add.

```
interface Loopback200
ipv4 address 20.0.0.1 255.255.255.0
pool vrf default ipv4 IPoE_POOL_BASE
 network 20.0.0.0/24 default-router 20.0.0.1
 exclude 20.0.0.0 20.0.0.253
```

But, testing shows that the addition of the secondary ipv4 address under loopback 200 cause all of the existing sessions being cleared automatically, since the Loopback 200 is unnumbered by all of the sessions and defined in a dynamic-template. This is not a smoothy in-service modification. To avoid this service
Optimising IP subscriber configuration for a BNG deployment

outage, we need to find a way to add the subnet only, without modify the unnumbered interface.

The configuration with optimisation-1 resolves this issue and also brings some other benefits comparing to original config.

---

**Original Config**

```plaintext
dhcp ipv4  
profile SERVER_BASE server  
 lease 0 0 10  
pool IPoE_POOL_BASE  
dns server 5.5.5.5 6.6.6.6  

! 
interface Bundle-Ether1.1001 server profile SERVER_BASE  
!  
pool vrf default ipv4 IPoE_POOL_BASE  
 network 20.0.0.0/24 default-router 20.0.0.1  
 exclude 20.0.0.0 20.0.0.253  
 exclude 21.0.0.0 21.0.0.253  

interface Bundle-Ether1.1001  
 ipv4 point-to-point  
 ipv4 unnumbered Loopback200  
 service-policy type control subscriber CP_IPOE_DHCPV4_BASE  
 encapsulation dot1q 1 second-dot1q 1001  
 ipscheduler ipv4 l2-connected  
 initiator dhcp  
!  
policy-map type control subscriber CP_IPOE_DHCPV4_BASE  
 event session-start match-first  
  class type control subscriber DHCPv4 do-until-failure  
   10 activate dynamic-template DTP_IPOE_DHCPV4  
!  
end-policy-map  
!  
dynamic-template  
 type ipscheduler DTP_IPOE_DHCPV4  
 ipv4 unnumbered Loopback200  
!  
interface Loopback200  
 description loopback for DHCP_triggered IP service  
 ipv4 address 20.0.0.1 255.255.255.0  
 ipv4 address 21.0.0.1 255.255.255.0 secondary
```

---

**Config with optimization-1**

```plaintext
dhcp ipv4  
profile SERVER_BASE server  
 lease 0 0 10  
pool IPoE_POOL_BASE  
subnet-mask 255.255.255.255  
default-router 100.100.100.100  
dns server 5.5.5.5 6.6.6.6  

! 
interface Bundle-Ether1.1001 server profile SERVER_BASE  
!  
pool vrf default ipv4 IPoE_POOL_OPT  
 address-range 20.0.0.1 20.0.0.254  
 address-range 21.0.0.1 21.0.0.254  
 exclude 20.0.0.0 20.0.0.253  
 exclude 21.0.0.0 21.0.0.253  

interface Bundle-Ether1.1001  
 ipv4 point-to-point  
 ipv4 unnumbered Loopback200  
 service-policy type control subscriber CP_IPOE_DHCPV4_BASE  
 encapsulation dot1q 1 second-dot1q 1001  
 ipscheduler ipv4 l2-connected  
 initiator dhcp  
!  
policy-map type control subscriber CP_IPOE_DHCPV4_BASE  
 event session-start match-first  
  class type control subscriber DHCPv4 do-until-failure  
   10 activate dynamic-template DTP_IPOE_DHCPV4  
!  
end-policy-map  
!  
dynamic-template  
 type ipscheduler DTP_IPOE_DHCPV4  
 ipv4 unnumbered Loopback200  
!  
interface Loopback200  
 description loopback for DHCP_triggered IP service  
 ipv4 address 100.100.100.100 255.255.255.255  
 ipv4 address 21.0.0.1 255.255.255.0 secondary
```
key points and the consideration of optimisation–1

- Defining the address-range in the pool, rather than defining network + default-router. at the same time, adding the configuration of default-router and subnet mask in the dhcp profile. With those configuration, no matter which address-range the IP is assigned from, the default-router assigned is fixed to 100.100.100.100, and the subnet mask is also defined explicitly as 255.255.255.255.

- As long as the default-router address and dhcp server identifier address is defined on the BNG(under a loopback and the sessions are unnumbered to that loopback), the BNG will reply to the ARP for that IP address. in this case, it’s 100.100.100.100.

- From 5.1.1, the address to which a session is unnumbered could be not necessarily in the same subnet where the DHCP client resides.

- Also please be noted, the address to which a access interface (bundle-e 1.1001) is unnumbered could be not necessarily in the same subnet where the DHCP client resides.

- In the testing, both APPLE and Mercury support to have the IP with /32 mask, and the default-router is in different subnet against the assigned IP.

- When add another address-range to the pool, the only CLI needed to add is address-range 21.0.0.1 21.0.0.254, no need to modify the loopback 200. the addition of new address-range cause no session/service interruption.

With above optimized configuration, the dhcp addressing result are summarized in table below. also paste the result for the original configuration here for comparison.

Addressing result with original config

<table>
<thead>
<tr>
<th>Client name</th>
<th>MAC address</th>
<th>You IP Address</th>
<th>subnet mask</th>
<th>DHCP Server identifier (option 54)</th>
<th>Router(option3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>3c07.545f.c041</td>
<td>20.0.0.254</td>
<td>255.255.255.0</td>
<td>20.0.0.1</td>
<td>20.0.0.1</td>
</tr>
<tr>
<td>Mercury</td>
<td>a857.4e06.4f47</td>
<td>21.0.0.254</td>
<td>255.255.255.0</td>
<td>20.0.0.1</td>
<td>21.0.0.1</td>
</tr>
</tbody>
</table>
Optimising IP subscriber configuration for a BNG deployment

Addressing result under Config with optimization_1

<table>
<thead>
<tr>
<th>Client name</th>
<th>MAC address</th>
<th>You IP Address</th>
<th>subnet mask</th>
<th>DHCP Server identifier (option 54)</th>
<th>Router(optim3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>3c07.545f.c041</td>
<td>20.0.0.254</td>
<td>255.255.255 .255</td>
<td>100.100.100 .100</td>
<td>100.100.100 .100</td>
</tr>
<tr>
<td>Mercury</td>
<td>a857.4e06.4f47</td>
<td>21.0.0.254</td>
<td>255.255.255 .255</td>
<td>100.100.100 .100</td>
<td>100.100.100 .100</td>
</tr>
</tbody>
</table>

The Apple and Mercury can ping each other, and here is the IP address assigned to the APPLE. Following is the display from MAC mini.
Another side benefit is that the number of ARP entry is half to that with secondary IP under the loopback 200. Since each virtual interface (created by session) has only one ARP entry for its local address now. Imagine if you have tens of thousands of sessions, this is a huge saving.

**ARP table with original config**

```
RP/0/RSP0/CPU0:Roy_BNG_1#sh arp
Wed Feb 12 00:29:31.051 UTC

0/0/CPU0

Address          Age        Hardware Addr   State      Type  Interface
20.0.0.1          -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001
20.0.0.1          -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001.ip1
20.0.0.1          -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001.ip2
21.0.0.1          -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001
21.0.0.1          -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001.ip1
21.0.0.1          -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001.ip2
```

**ARP table with optimisation-1 config**

```
RP/0/RSP0/CPU0:Roy_BNG_1#sh arp
Wed Feb 12 00:23:35.454 UTC

0/0/CPU0

Address          Age        Hardware Addr   State      Type  Interface
100.100.100.100   -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001
100.100.100.100   -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001.ip28
100.100.100.100   -          e4c7.223a.d1f3  Interface  ARPA  Bundle-Ether1.1001.ip29
```
Optimising IP subscriber configuration for a BNG deployment

Optimisation-2 Disabling the ARP learning

Disable the ARP learning/replying in the access-interface.

- block the ingress traffic from stale subscriber or IP spoofer

- Anti ARP spoofing

For sake of security

In the context of IPoE session deployment, there are two kinds of BAD GUY you need to pay attention to, Stale subscriber and IP spoofer.

Stale subscriber
A subscriber is called a Stale subscriber when it’s in following state. The session of this subscriber is torn/cleared on the BNG for some reason, but from the DHCP client’s point of view, it is still holding a valid IP address since the IP address is assigned by DHCP server and within a lease time. In this case the client is not aware of what happened on the BNG, and may keep sending IP packet to the BNG.

IP Spoofer
An IP spoofer is a subscriber who has it CPE configured with IP address and default-gateway address statically/manually. IP spoofer may also spoof the ARP.

It’s common between a stale subscriber and an IP spoofer that both of them has an IP address, but on the BNG, there is no subscriber session entry or subscriber route for them.

The difference could be that a stale subscriber knows the right default-gateway IP via DHCP, and may have the right MAC for the default-gateway cached in its ARP table. But an IP spoofer most likely does not have the right default-gateway IP, or even it has the right default-gateway IP, it still need to resolve it by ARP.

What the stale subscriber and IP spoofer can do harm to the BNG?

1, Invalid inbound traffic

Let’s firstly do some study on the outbound and inbound packet processing on ASR9K BNG.

Outbound traffic behaviour (Network→subscriber direction)
When an access-interface is unnumbered to an loopback interface, the connected route for the subnet is point to the loopback interface, rather than the access-interface. In this case, no traffic destined to the IP of a stale subscriber or an IP spoofer could be forwarded to it by the BNG, since there is no subscriber route for
Optimising IP subscriber configuration for a BNG deployment

it on the BNG. A subscriber route is a /32 route created by the BNG only when a session is created and in activate state, and would be deleted when a session is torn down.

In this mean, you do not need to worry that an IP spoofer/stale subscriber may either access to the internet via BNG(no reverse traffic), or hijack the reverse traffic bound for a valid subscriber.

**Inbound traffic behavior (Subscriber-to-Network direction)**

There is no IP source guard feature supported on ASR9K BNG, so when a IP packet is sent by the stale subscriber or IP spoofer arrives at the BNG access-interface, BNG would not check if the “source MAC and source IP binding” is valid against it’s DHCP binding table. Another word, the BNG would not drop any incoming packet as long as the destination MAC is itself correctly. BNG just forward it based on its forwarding table. The prerequisite is that the sender has the default gateway address resolved correctly.

A stale subscriber normally has the right MAC of the default–gateway cached in it’s ARP table before it’s ARP ages, but, for an IP spoofer, it need 1) to configure the correct default–gateway statically, and 2) resolve the MAC of default–gateway by sending ARP to the BNG and get ARP reply from BNG. Fortunately, there is a CLI on the BNG to prevent the BNG from replying the ARP request send by a host whose MAC address is not listed in it’s subscriber table. Another word, BNG can disable it’s ARP learning/replying for non–subscriber.

With the ARP replying disabled on the BNG, the IP spoofer is prevented from resolving the default–gateway MAC, hence it could not send IP traffic to the BNG.

**2, ARP spoofing**

With the ARP learning disabled( this is the other side of the coin) on the access–interface, you can prevent the ARP table on the BNG from been poisoned or spoofed.

Please be aware of the following facts of ASR9K BNG implementation.

1) When a session is triggered by the FSOL(in this context a DHCPv4 discover), on the BNG there is a virtual session–interface created dynamically for that session. The MAC of the host/CPE who triggers the session is processed as the session key, and all of the packet from/to that MAC address is treated as a packet over a virtual session–interface , rather than over the access–interface.

2) Packet from/to a MAC address which does not match any exist session MAC is treated as a packet over the access–interface.

3) Access–interface and session–interface has different interface handler, Although the session–interface is always beneath a access–interface, they can
Optimising IP subscriber configuration for a BNG deployment

have different behaviors and setting in terms of MTU, statistics counter, ARP, IP addressing, VRF, QoS, ACL etc. It’s important to distinct if a packet is handled by the access-interface or a session-interface.

### access-interface and virtual session-interface

```
RP/0/RSP0/CPU0:Roy_BNG_1#sh run int bundle-e 1.2000  <-- config for an access-intf
Fri Mar  7 16:21:41.659 UTC
interface Bundle-Ether1.2000
ipv4 point-to-point
ipv4 unnumbered Loopback200
arp learning disable
service-policy type control subscriber CP_IPOE_DHCPV4_SESSION_RESTART
ipssubscriber ipv4 l2-connected
  initiator dhcp
  initiator unclassified-source
!  encapsulation ambiguous dot1q 1 second-dot1q 2000-2010
```

```
RP/0/RSP0/CPU0:Roy_BNG_1#sh subscriber sess all  <-- a session created dynamically
Fri Mar  7 16:30:21.834 UTC
Codes: IN - Initialize, CN - Connecting, CD - Connected, AC - Activated,
       ID - Idle, DN - Disconnecting, ED - End

<table>
<thead>
<tr>
<th>Type</th>
<th>Interface</th>
<th>State</th>
<th>Subscriber IP Addr / Prefix</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BE1.2000.ip1</td>
<td>AC</td>
<td>20.0.0.254 (default)</td>
</tr>
</tbody>
</table>
```

```
RP/0/RSP0/CPU0:Roy_BNG_1#sh int b
Fri Mar  7 16:22:11.598 UTC

<table>
<thead>
<tr>
<th>Interface</th>
<th>IP-Address</th>
<th>Status</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>BVI1</td>
<td>172.18.88.1</td>
<td>Up</td>
<td>Up</td>
</tr>
<tr>
<td>Bundle-Ether1</td>
<td>unassigned</td>
<td>Up</td>
<td>Up</td>
</tr>
<tr>
<td>Bundle-Ether1.1</td>
<td>192.168.0.1</td>
<td>Up</td>
<td>Up</td>
</tr>
<tr>
<td>Bundle-Ether1.2</td>
<td>20.0.0.1</td>
<td>Shutdown</td>
<td>Down</td>
</tr>
</tbody>
</table>
| Bundle-Ether1.1001 | 100.100.100.100 | Up     | Up       | <-- access-interface
| Bundle-Ether1.2000 | 100.100.100.100 | Up     | Up       | <-- access-interface
| Bundle-Ether1.2000.ip1 | 100.100.100.100 | Up     | Up       | <<<< virtual session-interface
```
Optimising IP subscriber configuration for a BNG deployment

```
show interface for access-intf and session-intf

RP/0/RSP0/CPU0:Roy_BNG_1#sh int bundle-e 1.2000  <-- access-interface
Fri Mar  7 16:21:53.289 UTC
Bundle-Ether1.2000 is up, line protocol is up
  Interface state transitions: 1
  Hardware is VLAN sub-interface(s), address is e4c7.223a.d1f3
  Internet address is 100.100.100.100/32
  MTU 1522 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
    reliability 253/255, txload 0/255, rxload 0/255
  Encapsulation 802.1Q Virtual LAN,
    Outer Match: Dot1Q VLAN 1
    Inner Match: Dot1Q VLAN 2000-2010
  Ethertype Any, MAC Match src any, dest any
  loopback not set,
  ARP learning is disabled
  Last input 00:00:00, output 00:00:00
  Last clearing of "show interface" counters never
  5 minute input rate 0 bits/sec, 0 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
    46164 packets input, 4859378 bytes, 863 total input drops
    0 drops for unrecognized upper-level protocol
    Received 2 broadcast packets, 863 multicast packets
    45261 packets output, 5592024 bytes, 0 total output drops
    Output 1 broadcast packets, 0 multicast packets

RP/0/RSP0/CPU0:Roy_BNG_1#sh int bundle-e 1.2000.ip1  <-- session-interface
Fri Mar  7 16:22:01.934 UTC
Bundle-Ether1.2000.ip1 is up, line protocol is up
  Interface state transitions: 1
  Hardware is IP Subscriber interface(s), address is e4c7.223a.d1f3
  Internet address is 100.100.100.100/32
  MTU 1500 bytes, BW 10000000 Kbit (Max: 10000000 Kbit)
    reliability Unknown, txload Unknown, rxload Unknown
  Encapsulation Ipsub_base, loopback not set,
  ARP learning is disabled
  Last input 00:00:00, output 00:00:00
  Last clearing of "show interface" counters never
  45310 packets input, 4800620 bytes
    0 input drops, 0 queue drops, 0 input errors
    0 drops for unrecognized upper-level protocol
    45270 packets output, 5593194 bytes
    0 output drops, 0 queue drops, 0 output errors
```
4) On ASR9K BNG, the adjacency information for a host who connected to the BNG via an DHCP triggered session is not learned by ARP protocol, instead, it’s populated by the DHCP process. Acting as a DHCP proxy or DHCP server, the mapping between a MAC and the IP address assigned to it is naturally known by the BNG. BNG neither sends any ARP request over a virtual–interface, nor learns ARP over virtual–interface. But from the host’s point of view, it has to send ARP request to the BNG for the default–Gateway address resolution after the DHCP process completes, BNG will treat it as a ARP request over a session–interface, and reply to it.

5) An ARP request or Gratuitous ARP received over the access–interface sent by a IP spoofer or an stale subscriber may poison the ARP and adjacency table on the ASR9K BNG, imagine the case that it tells the BNG that “my MAC is MAC_B, and My IP address is IP_A”, but in fact that there is another valid session with MAC_A holding the IP_A. The result of the ARP poisoning may cause the traffic towards IP_A being sent to MAC_B, or just dropped by BNG since it’s Adjacency table is corrupted.

Conclusion

Invalid inbound traffic and ARP spoofing could only come over the access–interface.

By disabling the ARP learning on the access–interface, you can avoid the invalid inbound traffic in most cases, and eliminate the ARP–spoofing.

```conf
config with optimisation 2

interface Bundle-Ether1.1001
ipv4 point-to-point
ipv4 unnumbered Loopback200
arp learning disable
service-policy type control subscriber CP_IPOE_DHCPV4_BASE
encapsulation dot1q 1 second-dot1q 1001
ipsubscriber ipv4 l2-connected
initiator dhcp
```
BTW, in some case you may not want to disable the ARP learning over the access-interface totally, you can use "arp learning local" instead. This CLI make the access-interface only learn and response to the ARP from IP address resides in the same subnet to it's own interface IPv4 address.

A scenario that you don’t want to disable the ARP learning over the access-interface is that you want to use a access-interface to accommodate both IPoE session traffic and non-session traffic simultaneously, the later one is handled as traffic over access-interface. for example, you have in this case the session and non-session service will not share the same subnet. with "arp learning local" configured, you can have the ARP for non-session service learned and make sure that the adjacency for the session based CPE does not suffer ARP poisoning.

```bash
interface Bundle-Ether1.1001
  description an_example_for_mixed_session_nonsession
  ipv4 address 33.33.0.0/24
  arp learning local
  service-policy type control subscriber CP_IPOE_DHCPV4_BASE
  encapsulation dot1q 1 second-dot1q 1001
  ipsubscriber ipv4 l2-connected
  initiator dhcp

! the DHCP triggered session will get IP address from another subnet, for example 22.22.0.0/24
```
Optimising IP subscriber configuration for a BNG deployment

**Optimization-3 Disabling ip unreachable**

Disabling ip unreachable for both access-interface and session-interface.

For sake of security and scalability

When a session is established, if the CPE send packet to an destination IP to whom there is no route on the BNG, BNG will reply with “ICMP destination unreachable” message, which is handled by the RP CPU.

To avoid this unnecessary burden, you can disable the ip unreachable for for session-interface by configuration in dynamic-template. And you can also disable the ip unreachable on the access-interface.

```bash
config with optimisation 3

interface Bundle-Ether1.1001
ipv4 point-to-point
ipv4 unnumbered Loopback200
arp learning disable
ipv4 unreachables disable
service-policy type control subscriber CP_IPOE_DHCPV4_BASE
encapsulation dot1q 1 second-dot1q 1001
ipsubscriber ipv4 l2-connected
initiator dhcp

dynamic-template
type ipsubscriber DTP_IPOE_DHCPV4
ipv4 unnumbered Loopback200
ipv4 unreachables disable
```
Optimising IP subscriber configuration for a BNG deployment

---end of document---