Configuring ARP

Configuring ARP

Address resolution is the process of mapping network addresses to Media Access Control (MAC) addresses, which is typically done dynamically by the system using the ARP protocol, but can also be done by Static ARP entry configuration. This process is accomplished using the Address Resolution Protocol (ARP).

ARP is used to associate IP addresses with media or MAC addresses. Taking an IP address as input, ARP determines the associated media address. After a media or MAC address is determined, the IP address or media address association is stored in an ARP cache for rapid retrieval. Then the IP datagram is encapsulated in a link-layer frame and sent over the network.

ARP and Proxy ARP

Two forms of address resolution are supported by Cisco IOS XR software: Address Resolution Protocol (ARP) and proxy ARP, as defined in RFC 826 and RFC 1027, respectively. Cisco IOS XR software also supports a form of ARP called local proxy ARP.

For more details on Proxy ARP and Local Proxy ARP, see Proxy ARP and Local Proxy ARP, on page 4

Restrictions

The following restrictions apply to configuring ARP :

- Reverse Address Resolution Protocol (RARP) is not supported.
- ARP throttling, which is the rate limiting of ARP packets in Forwarding Information Base (FIB), is not supported.

Address Resolution Overview

A device in the IP can have both a local address (which uniquely identifies the device on its local segment or LAN) and a network address (which identifies the network to which the device belongs). The local address is more properly known as a *data link address*, because it is contained in the data link layer (Layer 2 of the OSI model) part of the packet header and is read by data-link devices (bridges and all device interfaces, for

example). The more technically inclined person will refer to local addresses as *MAC addresses*, because the MAC sublayer within the data link layer processes addresses for the layer.

To communicate with a device on Ethernet, for example, Cisco IOS XR software first must determine the 48-bit MAC or local data-link address of that device. The process of determining the local data-link address from an IP address is called address resolution.

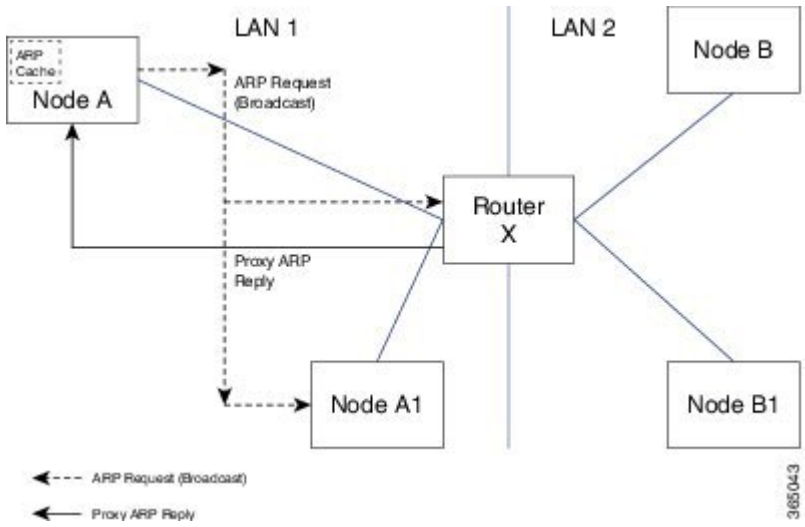
Address Resolution on a Single LAN

1. End System A (Node A) broadcasts an ARP request onto the LAN, attempting to learn the MAC address of End System B (Node B).
2. The broadcast is received and processed by all devices on the LAN, including End System B.
3. Only End System B replies to the ARP request. It sends an ARP reply containing its MAC address to End System A (Node A).
4. End System A (Node A) receives the reply and saves the MAC address of End System B in its ARP cache. (The ARP cache is where network addresses are associated with MAC addresses.)

   On learning the entry from End System B (Node B) adjacency is programmed in the hardware of system A (Node A). Further, packet forwarding for Node B is taken care by the hardware of system A (Node A).

Address Resolution When Interconnected by a Router

The following process describes address resolution when the source and destination devices are attached to different LANs that are interconnected by a router (only if proxy-arp is turned on):



1. End System Y (Node A) broadcasts an ARP request onto the LAN, attempting to learn the MAC address of End System Z (Node B).
2. The broadcast is received and processed by all devices on the LAN, including Router X.
3. Router X checks its routing table and finds that End System Z (Node B) is located on a different LAN.
4. Router X therefore acts as a proxy for End System Z (Node B). It replies to the ARP request from End System Y (Node A), sending an ARP reply containing its own MAC address as if it belonged to End System Z (Node B).

5. End System Y (Node A) receives the ARP reply and saves the MAC address of Router X in its ARP cache, in the entry for End System Z (Node B).
6. When End System Y (Node A) needs to communicate with End System Z (Node B), it checks the ARP cache, finds the MAC address of Router X, and sends the frame directly, without using ARP requests.
7. Router X receives the traffic from End System Y (Node A) and forwards it to End System Z (Node B) on the other LAN.

ARP Cache Entries

ARP establishes correspondences between network addresses (an IP address, for example) and Ethernet hardware addresses. A record of each correspondence is kept in a cache for a predetermined amount of time and then discarded.

You can also add a static (permanent) entry to the ARP cache that persists until explicitly removed.

Defining a Static ARP Cache Entry

ARP and other address resolution protocols provide a dynamic mapping between IP addresses and media addresses. Because most hosts support dynamic address resolution, generally you need not specify static ARP entries. If you must define them, you can do so globally. Performing this task installs a permanent entry in the ARP cache. Cisco IOS XR software uses this entry to translate 32-bit IP addresses into 48-bit hardware addresses.

Optionally, you can specify that the software responds to ARP requests as if the software was identified by the specified IP address, by making an alias entry in the ARP cache.

Configuration Example

A cache entry is created to establish connection between an IP address **203.0.113.2** and the MAC address **0010.9400.000c**. Additionally, the cache entry is created as an alias entry such that the interface to which the entry is attached will respond to ARP request packets for this network layer address with the data link layer address in the entry.

```
Router#config
Router(config)#arp 203.0.113.2 0010.9400.000c arPA
Router(config)#commit
```

Running Configuration

```
Router#show run arp 203.0.113.2 0010.9400.000c arpA
arp vrf default 203.0.113.2 0010.9400.000c ARPA
```

Verification

Verify that the State is static for proper functioning:

```
Router#show arp location 0/0/CPU0
Address         Age        Hardware Addr   State      Type  Interface
203.0.113.1       -          ea28.5f0b.8024  Interface  ARPA  HundredGigE0/0/0/9
203.0.113.2       -          0010.9400.000c  Static ARPA  HundredGigE0/0/0/9
```

Proxy ARP and Local Proxy ARP

When proxy ARP is disabled, the networking device responds to ARP requests received on an interface only if one of the following conditions is met:

- The target IP address in the ARP request is the same as the interface IP address on which the request is received.
- The target IP address in the ARP request has a statically configured ARP alias.

When proxy ARP is enabled, the networking device also responds to ARP requests that meet all the following conditions:

- The target IP address is not on the same physical network (LAN) on which the request is received.
- The networking device has one or more routes to the target IP address.
- All of the routes to the target IP address go through interfaces other than the one on which the request is received.

When local proxy ARP is enabled, the networking device responds to ARP requests that meet all the following conditions:

- The target IP address in the ARP request, the IP address of the ARP source, and the IP address of the interface on which the ARP request is received are on the same Layer 3 network.
- The next hop for the target IP address is through the same interface as the request is received.

Typically, local proxy ARP is used to resolve MAC addresses to IP addresses in the same Layer 3 network. Local proxy ARP supports all types of interfaces supported by ARP and unnumbered interfaces.

Enabling Proxy ARP

Cisco IOS XR software uses proxy ARP (as defined in RFC 1027) to help hosts with no knowledge of routing determine the media addresses of hosts on other networks or subnets. For example, if the router receives an ARP request for a host that is not on the same interface as the ARP request sender, and if the router has all of its routes to that host through other interfaces, then it generates a proxy ARP reply packet giving its own local data-link address. The host that sent the ARP request then sends its packets to the router, which forwards them to the intended host. Proxy ARP is disabled by default; this task describes how to enable proxy ARP if it has been disabled.

Configuration Example

Proxy ARP is enabled on the HundredGigE interface-0/0/0/0:

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/0
Router(config-if)#proxy-arp
Router(config-if)#commit
```

Running Configuration

```
Router# show running-config interface HundredGigE 0/0/0/0
interface HundredGigE0/0/0/0
 mtu 4000
 ipv4 address 192.0.2.1 255.255.255.0
 proxy-arp
```

```
!
!
```

Verification

Verify that proxy ARP is configured and enabled:

```
Router#show arp idb HundredGigE 0/0/0/0 location 0/0/CPU0
  interface HundredGigE0/0/0/0 (0x08000038):
  IPv4 address 192.0.2.1, Vrf ID 0x60000000
  VRF Name default
  Dynamic learning: Enable
  Dynamic entry timeout: 14400 secs
  Purge delay: off
  IPv4 caps added (state up)
  MPLS caps not added
  Interface not virtual, not client fwd ref,
  Proxy arp is configured, is enabled
  Local Proxy arp not configured
  Packet IO layer is NetIO
  Srg Role : DEFAULT
  Idb Flag : 262332
  IDB is Complete
```

Enabling Local Proxy ARP

Local proxy ARP is used to resolve MAC addresses to IP addresses in the same Layer 3 network such as, private VLANs that are Layer 2-separated. Local proxy ARP supports all types of interfaces supported by ARP and unnumbered interfaces.

Configuration Example

Local proxy ARP is enabled on the HundredGigE interface-0/0/0/0

```
Router#configure
Router(config)#interface HundredGigE 0/0/0/0
Router(config-if)#local-proxy-arp
Router(config-if)#commit
```

Running Configuration

```
Router#show running-config interface HundredGigE 0/0/0/0
 interface HundredGigE0/0/0/0
ipv4 address 192.0.2.1 255.255.255.0
local-proxy-arp
!
```

Verification

Verify that local proxy ARP is configured:

```
Router#show arp idb HundredGigE0/0/0/0 location 0/0/CPU0
HundredGigE0/0/0/0(0x08000038):
  IPv4 address 192.0.2.1, Vrf ID 0x60000000
  VRF Name default
  Dynamic learning: Enable
  Dynamic entry timeout: 14400 secs
  Purge delay: off
  IPv4 caps added (state up)
  MPLS caps not added
  Interface not virtual, not client fwd ref,
```

```
  Proxy arp not configured, not enabled
  Local Proxy arp is configured
  Packet IO layer is NetIO
  Srg Role : DEFAULT
  Idb Flag : 264332
  IDB is Complete
```

Associated Commands

- local-proxy-arp
- show arp idb

Configuring ARP purge-delay

With Equal Cost Multi Path (ECMP), traffic is load balanced across multiple paths with equal cost. This should provide resiliency against interface flaps. If an interface goes down, the traffic is then routed via the other interface without traffic loss. However, if the first interface comes up, traffic is routed back over it but forwarding will only resume once ARP has been (re)resolved and the adjcency (re)installed. Here a short unexpected interface flap causes this traffic loss and is particularly undesirable.

The purge-delay feature allows existing dynamic entries to persist rather than immediately delete entries which could cause traffic loss following an interface flap.

The purge delay feature works by caching existing dynamic ARP entries when an interface goes down and starting a purge delay timer. When the interface is brought back and the purge delay timer not yet fired, the entries are reinstalled as before. The normal entry timeout is reduced in order to re-ARP for the entries after any interface state change related churn has died down; should the purge delay timer fire before the interface comes back up, the entries are deleted from the cache.

Configuring Example

```
Router(config)# interface HundredGigE 0/0/0/34
Router(config-if)# arp purge-delay 100
Router(config-if)# commit
```

Running Configuration

```
Router#show running-config interface HundredGigE 0/0/0/34
Wed Jul 24 09:14:28.200 UTC
interface HundredGigE0/0/0/34
 arp purge-delay 100
 shutdown
!
```

Verification

```
Router#show arp idb HundredGigE 0/0/0/34 location 0/RP0/CPU0
Wed Jul 24 09:16:16.593 UTC

HundredGigE0/0/0/34 (0x0f000208):
  IDB Client: default
  IPv4 address 19.0.2.1, Vrf ID 0x00000000
  VRF Name unknown
  Dynamic learning: Enable
  Dynamic entry timeout: 14400 secs
  Drop adjacency timeout: Disable
  Purge delay: 100 seconds
```

```
  IPv4 caps not added
  MPLS caps not added
  Interface not virtual, not client fwd ref,
  Proxy arp not configured, not enabled
  Local Proxy arp not configured
  Packet IO layer is NetIO
  Srg Role : DEFAULT
  Idb Flag : 0
  IDB is not Complete
, No Media, No HW Addr, No IPv4 Caps, No IPv4 State, No IPv4 Addr, No VRF

 Total entries : 1
| Event Name            |     Time Stamp       | S, M
| idb-update               | Jul 24 09:13:42.912  | 1, 0
```

Configuring ARP Timeout

Dynamic ARP entries which are learnt by ARP address resolution (when valid ARP replies are received) are timed out every 4 hours by default in order to remove stale entries.

ARP entries that correspond to the local interface or that are statically configured by the user never time out.

Configuring Example

```
Router(config)# interface HundredGigE 0/0/0/35
Router(config-if)# arp timeout 100
Router(config-if)# commit
```

Running Configuration

```
Router#show running-config interface HundredGigE 0/0/0/35
Wed Jul 24 08:56:56.428 UTC
interface HundredGigE0/0/0/35
 arp timeout 100
 shutdown
!
```

Verification

```
Router#show arp idb HundredGigE 0/0/0/35 location 0/RP0/CPU0
Wed Jul 24 09:04:55.127 UTC

HundredGigE0/0/0/35 (0x0f000200):
  IDB Client: default
  IPv4 address 192.0.2.1, Vrf ID 0x00000000
  VRF Name unknown
  Dynamic learning: Enable
  Dynamic entry timeout: 100 secs
  Drop adjacency timeout: Disable
  Purge delay: off
  IPv4 caps not added
  MPLS caps not added
  Interface not virtual, not client fwd ref,
  Proxy arp not configured, not enabled
  Local Proxy arp not configured
  Packet IO layer is NetIO
  Srg Role : DEFAULT
  Idb Flag : 0
  IDB is not Complete
```

```
, No Media, No HW Addr, No IPv4 Caps, No IPv4 State, No IPv4 Addr, No VRF

 Total entries : 1
| Event Name            |     Time Stamp       | S, M
| idb-update              | Jul 24 08:56:13.440  | 1, 0
```

Limit ARP Cache Entries per Interface

Table 1: Feature History Table

Feature Name	Release Information	Feature Description
Limit Address Resolution Protocol (ARP) Cache Entries per Interface	Release 7.9.1	In this feature, you can configure the maximum limit for the number of entries of dynamic mapping between IP addresses and media addresses by ARP per interface. Limiting the number of entries provides overflow protections in ARP cache and protects the routers from DOS attacks by preventing memory overuse by cache entries. This feature introduces the **arp cache-limit** command.

The ARP cache overflow occurs when the number of entries in the cache exceeds the maximum limit value of 127999. Such instances make the router vulnerable to threats like DOS attacks. With this feature, you can configure the maximum limit of dynamic ARP entries learned per interface. The router won't accept any cache entries unless cleared after the number entries exceeds the maximum limit in the configuration. You can configure the maximum limit range of 0–127999 per interfaces in the router.

Note The arp cache resources vary depending on the hardware resources available in a router. Ensure the cache-limit configured such that the available resources in the router are able to accomodate the entries.

Feature highlights

This section details the good to know information for using ARP overflow protection:

- The router drops new ARP requests when the number of entries are more than or equal to the applied cache limit value.
- The router won't learn from ARP packets received after exceeding the applied cache limit value.
- The ARP cache limit isn't applicable to static ARP entries.
- The router doesn't enforce the ARP cache limit on ARP client triggered entries.
- The router issues a syslog message when it reaches the cache limit. For every 1000 entries after the cache limit, the router issues a new syslog message. The syslog message includes the interface name and cache

entries drop counters. For example, `RP/0/RP0/CPU0:Jul 1 10:10:25.781 IST: grid_svr[211]: %L2-GRID-4-BANK_FULL : GRID POOL:GLIF(2), BANK 0 FULL. Max size 4091, Curr RIDs 4091.`

- You can view the ARP entries statistics using the **show arp idb** command.
- The ARP Cache limit doesn't drops the already learned dynamic ARP entries. That is, if the number of dynamic ARP entries in the cache is higher or equal to the newer cache limit set in the router, then the router will neither take any new entries or drop the preexisting entries in the cache, but it will start issuing the syslog message the cache limit.

Configuration Example

The following example shows how to set the ARP cache limit for an interface:

Configuration

```
Router# configure
Router(config)# interface HundredGigE 0/0/0/0
Router(config-if)#arp cache-limit 3900
Router(config-if)#commit
```

Running Configuration

```
Router# show running-config interface HundredGigE 0/0/0/0
interface HundredGigE0/0/0/0
 arp cache-limit 3900
 !
!
```

Verification

```
Router#show arp idb HundredGigE 0/0/0/0 location RP0
HundredGigE (0x00000090):
  IDB Client: default
  IPv4 address 1.1.1.1, Vrf ID 0x60000000
  VRF Name default
  Dynamic learning: Enable
  Dynamic entry timeout: 14400 secs
  Drop adjacency timeout: Disable
  Purge delay: off
  Cache limit: 3900
  Incomplete glean count: 0
  Complete glean count: 0
  Complete protocol count: 0
  Dropped glean count: 0
  Dropped protocol count: 0
  IPv4 caps added (state up)
  MPLS caps not added
  Interface is virtual, not client fwd ref,
  Proxy arp not configured, not enabled
  Local Proxy arp not configured
  Packet IO layer is SPIO
  Srg Role : DEFAULT
  Idb Flag : 49294
  IDB is Complete
  IDB Flag Description:
  [VIRTUAL | CAPS | COMPLETE | IPV4_CAPS_CREATED |
   SPIO_ATTACHED | SPIO_SUPPORTED]
  Idb Flag Ext : 0x0
  Idb Oper Progress : NONE
  Client Resync Time : N/A
```

Direct Attached Gateway Redundancy

Direct Attached Gateway Redundancy (DAGR) allows third-party redundancy schemes on connected devices to use gratuitous ARP as a failover signal, enabling the ARP process to advertise a new type of route in the Routing Information Base (RIB). These routes are distributed by Open Shortest Path First (OSPF).

Sometimes part of an IP network requires redundancy without routing protocols. A prime example is in the mobile environment, where devices such as base station controllers and multimedia gateways are deployed in redundant pairs, with aggressive failover requirements (subsecond or less), but typically do not have the capability to use native Layer 3 protocols such as OSPF or Intermediate System-to-Intermediate System (IS-IS) protocol to manage this redundancy. Instead, these devices assume they are connected to adjacent IP devices over an Ethernet switch, and manage their redundancy at Layer 2, using proprietary mechanisms similar to Virtual Router Redundancy Protocol (VRRP). This requires a resilient Ethernet switching capability, and depends on mechanisms such as MAC learning and MAC flooding.

DAGR is a feature that enables many of these devices to connect directly to without an intervening Ethernet switch. DAGR enables the subsecond failover requirements to be met using a Layer 3 solution. No MAC learning, flooding, or switching is required.

Note Since mobile devices' 1:1 Layer 2 redundancy mechanisms are proprietary, they do not necessarily conform to any standard. So although most IP mobile equipment is compatible with DAGR, interoperability does require qualification, due to the possibly proprietary nature of the Layer 2 mechanisms with which DAGR interfaces.

Restrictions

The following additional restrictions apply when configuring the Direct Attached Gateway Redundancy (DAGR) feature:

- IPv6 is not supported.
- Ethernet bundles are not supported.
- Non-Ethernet interfaces are not supported.
- Hitless ARP Process Restart is not supported.
- Hitless RSP Failover is not supported.

Configuring DAGR

Configuration Example

```
Router# configure terminal
Router(config)# interface HundredGigE 0/0/0/1
Router(config-if)# arp dagr
Router(config-if-dagr)# peer ipv4 192.0.2.1
Router(config-if-dagr-peer)# route distance normal 140 priority 3
Router(config-if-dagr-peer)# route metric normal 84 priority 80
Router(config-if-dagr-peer)# timers query 2 standby 19
Router(config-if-dagr-peer)# priority-timeout 25
Router(config-if-dagr)# commit
```

Running Configuration

```
configure
 interface HundredGigE 0/0/0/1
  arp dagr
   peer ipv4 192.0.2.1
    priority-timeout 25
    route distance normal 140 priority 3
    route metric normal 84 priority 80
    timers query 2 standby 19
    commit
```

Verification

The following example shows how to display the current operational state of the DAGR groups:

```
Router# show arp dagr
------------------------------------------------------------------------------
0/1/CPU0
------------------------------------------------------------------------------
Interface                 Virtual IP      State     Query-pd Dist Metr
HundredGigE0/0/0/1     192.0.2.1    Active    None     150  100
HundredGigE0/0/0/1     192.0.2.45     Query     1        None None
```