Configuring ARP

- Configuring ARP, on page 1
- Information About Configuring ARP, on page 8

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.

For more details on ARP, see Information About Configuring ARP, on page 8

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 2

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.

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 **20.0.0.2** and the MAC address **20.0.2**. 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 20.0.0.2 20.0.2 arpA alias
```

Running Configuration

```
Router# show running-config interface tenGigE 0/0/0/0
arp 1.0.0.2 1.0.2 arpa
arp vrf default 1.0.0.2 0001.0000.0002 ARPA
```

Verification

Verify that the State is static for proper functioning:

```
Router# show arp 20.0.0.2
------------------------------------------------------------------------------
0/RP0/CPU0
------------------------------------------------------------------------------
Address         Age        Hardware Addr   State        Type  Interface
20.0.0.2        -          0020.0000.0002  Static ARPA  TenGigE0/0/0/0
```

Related Topics

Defining a Static ARP Cache Entry, on page 2

Associated Commands

- arp
- show arp

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 TenGigE interface-**0/0/0/0**:

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

Running Configuration

```
Router# show running-config interface tenGigE 0/0/0/0
interface TenGigE0/0/0/0
 mtu 4000
 ipv4 address 1.0.0.1 255.255.255.0
 proxy-arp
!
!
```

Verification

Verify that proxy ARP is configured and enabled:

```
Router# show arp idb tenGigE 0/0/0/0 location 0/RP0/CPU0
  TenGigE0/0/0/0 (0x08000038):
  IPv4 address 1.0.0.1, 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
```

Related Topics

- Enabling Proxy ARP, on page 3

Associated Commands

- proxy-arp

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 TenGigE interface-**0/0/0/0**

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

Running Configuration

```
Router# show running-config interface tenGigE 0/0/0/0
interface TenGigE0/0/0/0
ipv4 address 1.0.0.1 255.255.255.0
local-proxy-arp
!
```

Verification

Verify that local proxy ARP is configured:

```
Router# show arp idb tenGigE 0/0/0/0 location 0/RP0/CPU0
TenGigE0/0/0/0 (0x08000038):
  IPv4 address 1.0.0.1, 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

Configure Learning of Local ARP Entries

You can configure an interface or a sub-interface to learn only the ARP entries from its local subnet.

Use the following procedure to configure local ARP learning on an interface.

1. Enter the interface configuration mode.

```
Router(config)# interface GigabitEthernet 0/0/0/1
```

2. Configure the IPv4/IPv6 address for the interface.

```
Router(config-if)# ipv4 address 12.1.3.4 255.255.255.0
```

3. Configure local ARP learning on the interface.

```
Router(config-if)# arp learning local
```

4. Enable the interface and commit your configuration.

```
Router(config-if)# no shut
Router(config-if)# commit
RP/0/0/CPU0:Dec 12 13:41:16.580 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : interface
GigabitEthernet 0/0/0/1, changed state to Down
RP/0/0/CPU0:Dec 12 13:41:16.683 : ifmgr[397]: %PKT_INFRA-LINK-3-UPDOWN : interface
GigabitEthernet 0/0/0/1 changed state to Up
```

5. Confirm your configuration.

```
Router(config-if)# show running-configuration
..
Building configuration...
!! IOS XR Configuration 0.0.0
!! Last configuration change at Mon Dec 12 13:41:16 2016
!interface GigabitEthernet 0/0/0/1
 ipv4 address 12.1.3.4 255.255.255.0
 arp learning local
!
```

6. Verify if local ARP learning is working as configured on the interface.

```
Router(config-if)# do show arp idb gigabitEthernet 0/0/0/1 location 0/0/CPU0
Thu Dec 15 10:00:11.733 IST

GigabitEthernet 0/0/0/1 (0x00000040):
```

```
  IPv4 address 12.1.3.4, Vrf ID 0x60000000
  VRF Name default
  Dynamic learning: Local
  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 not configured
  Packet IO layer is NetIO
  Srg Role : DEFAULT
  Idb Flag : 2146444
  IDB is Complete
```

7. (Optional) You can monitor the ARP traffic on the interface.

```
Router(config-if)# do show arp idb gigabitEthernet 0/0/0/1 location 0/0/CPU0
Thu Dec 15 10:13:28.964 IST

ARP statistics:
  Recv: 0 requests, 0 replies
  Sent: 0 requests, 1 replies (0 proxy, 0 local proxy, 1 gratuitous)
  Subscriber Interface:
         0 requests recv, 0 replies sent, 0 gratuitous replies sent
  Resolve requests rcvd: 0
  Resolve requests dropped: 0
  Errors: 0 out of memory, 0 no buffers, 0 out of sunbet

ARP cache:
  Total ARP entries in cache: 1
  Dynamic: 0, Interface: 1, Standby: 0
  Alias: 0,   Static: 0,    DHCP: 0

  IP Packet drop count for GigabitEthernet0_0_0_1: 0
```

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.

Enabling DAGR

Configuration Example

DAGR is enabled on the TenGigE interface 0/0/0/10 with the peer router having ipv4 address of 5.10.10.1. The failover attributes set are priority-timeout, route distance, route metric, and timers query.

```
Router#configure
Router(config)#interface TenGigE 0/0/0/10
outer(config-if)#arp dagr
Router(config-if-dagr)#peer ipv4 5.10.10.1
Router(config-if-dagr-peer)#priority-timeout 1
Router(config-if-dagr-peer#route distance normal 1 priority 1
Router(config-if-dagr-peer#route metric normal 1 priority 1
Router(config-if-dagr-peer#timers query 2 standby 1
Router(config-if-dagr-peer#commit
```

Running Configuration

```
Router# show running-config interface tenGigE 0/0/0/10
mtu 9216
 ipv4 address 10.0.0.1 255.255.255.0
 arp dagr
  peer ipv4 1.0.0.1
   route metric normal 1 priority 1
   timers query 2 standby 1
   priority-timeout 1
   route distance normal 1 priority 1
  !
  peer ipv4 10.0.0.3
  !
 !
!
!
```

Verification

```
Router#show arp dagr
------------------------------------------------------------------------------
0/RP0/CPU0
------------------------------------------------------------------------------
Interface                 Virtual IP      State     Query-pd Dist Metr
TenGigE0/0/0/10           1.0.0.1         Standby   1        None None
TenGigE0/0/0/10           10.0.0.3        Query     1        None None
```

Related Topics

Direct Attached Gateway Redundancy, on page 6

Associated Commands

- arp dagr
- route distance
- route metric
- priority-timeout
- peer (DAGR)
- show arp dagr

Information About Configuring ARP

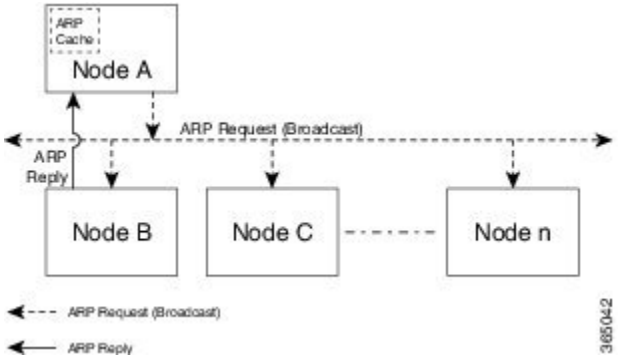
Addressing Resolution Overview

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

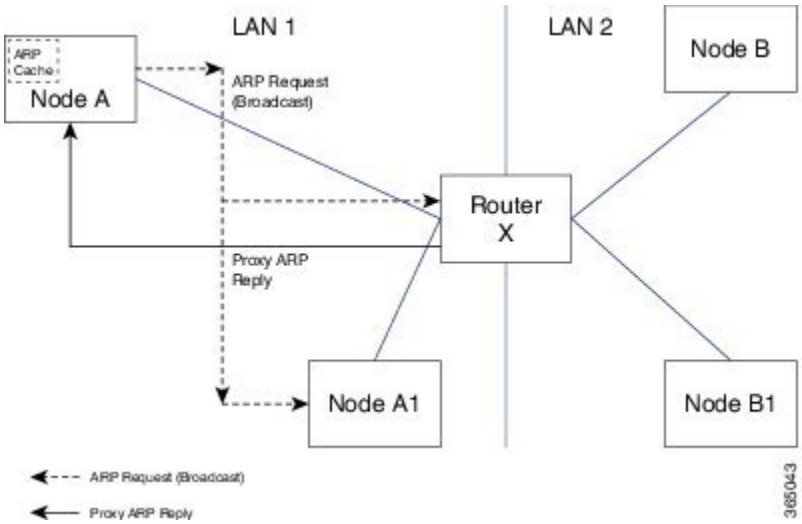
The following process describes address resolution when the source and destination devices are attached to the same LAN:



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.)
5. Whenever End System A (Node A) needs to communicate with End System B, it checks the ARP cache, finds the MAC address of System B, and sends the frame directly, without needing to first use an ARP request.

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.