ARP Overview, Standards and History

Physical networks function at layers one and two of the OSI Reference Model, and use data link layer addresses. In contrast, internetworking protocols function at layer three, interconnecting these physical networks to create a possibly huge internetwork of devices specified using network layer addresses. Address resolution is the process where network layer addresses are resolved into data link layer addresses, to permit data to be sent one hop at a time across an internetwork. I describe address resolution in detail in the preceding section on concepts.
The problem of address resolution was apparent from the very start in the development of the TCP/IP protocol suite. Much of the early development of IP was performed on the then-fledgling Ethernet local area networking technology; this was even before Ethernet had been officially standardized as IEEE 802.3. It was necessary to define a way to map IP addresses to Ethernet addresses to allow communication over Ethernet networks.

There are two basic methods that resolution could have been used to accomplish this correlation of addresses: direct mapping or dynamic resolution. However, Ethernet addresses are 48 bits long while IP addresses are only 32 bits, which immediately rules out direct mapping. Furthermore, the designers of IP wanted the flexibility that results from using the dynamic resolution model. To this end, they developed the TCP/IP *Address Resolution Protocol (ARP)*. This protocol is described in one of the earliest of the Internet RFCs still in common use: RFC 826, *An Ethernet Address Resolution Protocol*, published in 1982.

The name makes clear that ARP was originally developed for Ethernet. Thus, it represents a nexus between the most popular layer two LAN protocol and the most popular layer three internetworking protocol—this is true even two decades later. However, it was also obvious from the beginning that even if Ethernet was a very common way of transporting IP, it would not be the only one. Therefore, ARP was made a general protocol capable of resolving addresses from IP to not just Ethernet but numerous other data link layer technologies.

The basic operation of ARP involves encoding the IP address of the intended recipient in a broadcast message. It is sent on a local network to allow the intended recipient of an IP datagram to respond to the source with its data link layer address. This is done using a simple request/reply method described in the following topic on general operation. A special format is used for ARP messages, which are passed down to the local data link layer for transmission.

Key Concept: *ARP* was developed to facilitate dynamic address resolution between IP and Ethernet, and can now be used on other layer two technologies as well. It works by allowing an IP device to send a broadcast on the local network, requesting that another device on the same local network respond with its hardware address.

This basic operation is supplemented by methods to improve performance. Since it was known from the start that having to perform a resolution using broadcast for each datagram was ridiculously inefficient, ARP has always used a cache, where it keeps bindings between IP addresses and data link layer addresses on the local network. Over time, various techniques have been developed to improve the methods used for maintaining cache entries. Refinements and additional features, such as support for cross-resolution by pairs of devices as well as proxy ARP, have also been defined over the years and added to the basic ARP feature set.

ARP Address Specification and General Operation

(Page 1 of 2)

An Address Resolution Protocol transaction begins when a source device on an IP network has an IP datagram to send. It must first decide whether the destination device is on the local network or a distant network. If the former, it will send directly to the destination; if the latter, it will send the datagram to one of the routers on the physical network for forwarding. Either way, it will determine the IP address of the device that needs to be the immediate destination of its IP datagram on the local network. After packaging the datagram it will pass it to its ARP software for address resolution.

Basic operation of ARP is a *request/response* pair of transmissions on the local network. The source (the one that needs to send the IP datagram) transmits a broadcast containing information about the destination (the intended recipient of the datagram). The destination then responds unicast back to the source, telling the source the hardware address of the destination.

ARP Message Types and Address Designations

The terms source and destination apply to the same devices throughout the transaction. However, there are two different messages sent in ARP, one from the source to the destination and one from the destination to the source. For each ARP message, the *sender* is the one that is transmitting the message and the *target* is the one receiving it. Thus, the identity of the sender and target change for each message:

- **Request:** For the initial request, the sender is the source, the device with the IP datagram to send, and the target is the destination.

- **Reply:** For the reply to the ARP request, the sender is the destination; it replies to the source, which becomes the target.

It's a bit confusing, but you'll get used to it. ☺ Each of the two parties in any message has two addresses (layer two and layer three) to be concerned with, so four different addresses are involved in each message:

- **Sender Hardware Address:** The layer two address of the sender of the ARP message.

- **Sender Protocol Address:** The layer three (IP) address of the sender of the ARP message.

- **Target Hardware Address:** The layer two address of the target of the ARP message.

- **Target Protocol Address:** The layer three (IP) address of the target.

These addresses each have a position in the ARP message format.

ARP General Operation

With that background in place, let's look at the steps followed in an ARP transaction (which are also shown graphically in the illustration in Figure 48):

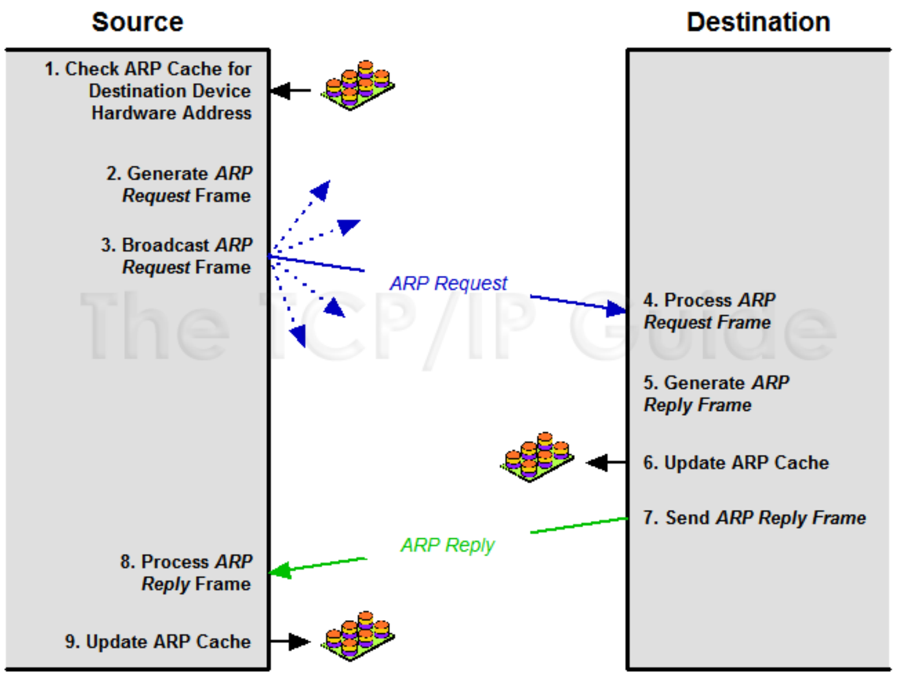
Source
Destination
1. Check ARP Cache for Destination Device Hardware Address
2. Generate ARP Request Frame
3. Broadcast ARP Request Frame
ARP Request
4. Process ARP Request Frame
5. Generate ARP Reply Frame
6. Update ARP Cache
7. Send ARP Reply Frame
ARP Reply
8. Process ARP Reply Frame
9. Update ARP Cache
The TCP/IP Guide

Figure 48: Address Resolution Protocol (ARP) Transaction Process

This diagram shows the sequence of steps followed in a typical ARP transaction, as well as the message exchanges between a source and destination device, and cache checking and update functions. (Those little columns are supposed to be hard disks, not cans of soup! ☺)

1. **Source Device Checks Cache:** The source device will first check its cache to determine if it already has a resolution of the destination device. If so, it can skip to the last step of this process, step #9.

2. **Source Device Generates ARP Request Message:** The source device generates an *ARP Request* message. It puts its own data link layer address as the *Sender Hardware Address* and its own IP address as the *Sender Protocol Address*. It fills in the IP address of the destination as the *Target Protocol Address*. (It must leave the *Target Hardware Address* blank, since that it is what it is trying to determine!)

3. **Source Device Broadcasts ARP Request Message:** The source broadcasts the *ARP Request* message on the local network.

4. **Local Devices Process ARP Request Message:** The message is received by each device on the local network. It is processed, with each device looking for a match on the *Target Protocol Address*. Those that do not match will drop the message and take no further action.

5. **Destination Device Generates ARP Reply Message:** The one device whose IP address matches the contents of the *Target Protocol Address* of the message will generate an *ARP Reply* message. It takes the *Sender Hardware Address* and *Sender Protocol Address* fields from the ARP Request message and uses these as the values for the *Target Hardware Address* and *Target Protocol Address* of the reply. It then fills in its own layer two address as the *Sender Hardware Address* and its IP address as the *Sender Protocol Address*. Other fields are filled in as explained in the topic describing the ARP message format.

6. **Destination Device Updates ARP Cache:** If the source needs to send an IP datagram to the destination now, it makes sense that the destination will probably need to send a response to the source at some point soon. (After all, most communication on a network is bidirectional.) As an optimization, then, the destination device will add an entry to its own ARP cache containing the hardware and IP addresses of the source that sent the ARP Request. This saves the destination from needing to do an unnecessary resolution cycle later on.

7. **Destination Device Sends ARP Reply Message:** The destination device sends the *ARP reply* message. This reply is, however, sent unicast to the source device, as there is no need to broadcast it.

8. **Source Device Processes ARP Reply Message:** The source device processes the reply from the destination. It stores the *Sender Hardware Address* as the layer two address of the destination, to use for sending its IP datagram.

9. **Source Device Updates ARP Cache:** The source device uses the *Sender Protocol Address* and *Sender Hardware Address* to update its ARP cache for use in the future when transmitting to this device.

Key Concept: ARP is a relatively simple request/reply protocol. The source device broadcasts an *ARP Request* looking for a particular device based on its IP address. That device responds with its hardware address in an *ARP Reply* message.

Note that this description goes a bit beyond the basic steps in address resolution, because two enhancements are mentioned. One is caching, which is described in its own topic but had to be mentioned here because it is the first step in the process, for obvious reasons. The other is cross-resolution (described in the overview of caching issues in dynamic resolution), which is step #6 of the process. This is why the source device includes its IP address in the request. It isn't really needed for any other reason, so you can see that this feature was built into ARP from the start.