# A Protocol for the Dynamic Assignment of IP Addresses for Use on an Ethernet

**Jeffrey I. Schiller**
**Mark A. Rosenstein**
***Massachusetts Institute of Technology***

## Abstract

This document describes a simple protocol to perform dynamic IP address assignment on an Ethernet network. The protocol is adaptable to be useful on any networking technology that supports broadcast or multicast and uses the ARP protocol (ARP is described in RFC 826).

## Problem

The number of single user workstations that use the IP/TCP protocol suite is growing. More importantly more of these workstations are being installed by ‘‘end users’’, people who are not networking experts. One problem that results from this is that not all workstations that are setup by non-experts get correctly configured to use the network. This can create problems both for the user of the improperly configured workstations, and more importantly it can create problems for other users on the network. The two typical parameters that have to be configured into a workstation for it to function at all on the network are its own IP address and the IP addresses of the gateways off this subnet. Other parameters are required to be a good neighbor on the network, such as the subnet mask and broadcast address.

The assignment of the workstation’s IP address presents the thorniest problem because the manual that comes with the workstation cannot tell you your IP address. It is necessary for the user to contact an appropriate (and typically different for each Ethernet) administrative person in order to get a uniquely assigned IP address. A related issue is to get a hostname for the workstation which matches the chosen address and is known the the proper nameservers.

## Requirements

A protocol to dynamically assign IP addresses must be capable of operating on a network that also has on it other workstations/hosts that do not participate in the address assignment protocol, without the addresses in use by those hosts being usurped.

The protocol should operate as independently from human interaction as possible. In other words a user of a workstation should not have to customize his workstation in any way. If any network configuration commands are used at all they should be of the form ‘‘network on’’ and ‘‘network off’’ with the protocol handling all other details.

Therefore the protocol must be capable of transmitting the following information to the workstation (during a time when the workstation does not yet have an IP address):

- The network number of the Ethernet.
- The addresses of the gateways off this Ethernet.

- The list of addresses that are permanently assigned to hosts that do not participate in the protocol.

The protocol should not rely on a sole information providing host, as this introduces a single point of failure.

## Proposed protocol

This protocol operates at the Ethernet (or 802.3) level of the network. There are two pieces. The first piece can be called the ‘‘Network Information Protocol’’ and is responsible for providing the workstation with the information it will need to configure itself. Armed with this information the workstation can then make a first stab at issuing itself an IP address. The second part of the protocol involves the use of the ARP protocol. The workstation attempts to use the ARP protocol to resolve an Ethernet address for the IP address it intends to use. If after a couple resends of the ARP resolve request it does not receive an answer, then it assumes that the IP address it picked is available for use and configures that address as its own IP address.

It is important that the workstation pick an IP address that is not likely to be picked by another workstation.[1] To facilitate this there are two ways that a workstation may choose its first ‘‘guess’’ at an IP address. The Network Information Protocol may provide the invoking workstation with a first guess as to what its IP address should be (more on this later). If this information is provided the workstation should use this as a first guess. If this information is not provided then the host should construct its own first guess using the network address, network mask and allowed range of assignable addresses to construct a likely to be unique host portion.

A hash function that is fed the Ethernet address of the workstation as a seed should suffice for this. It is likely that many of the ethernet boards on a subnet will be made by the same manufacturer, hence the first three bytes of their physical addresses will be the same. The remaining three bytes may be similar, but probably not actually successive numbers. It is sufficient for the address hash function to sum the last three bytes of the physical address, and take that number modulo the size of the allowable address range, and finally add this to the start of the allowable range.

It is important to understand that this arrives only at a first guess of what IP address the workstation should use. If that address is determined to be in use by using ARP as described above, the workstation should construct a second attempt and try again. The hash function should be modified as follows: Rather than simply incrementing the address and trying again, add the last byte of the physical address to the sum, and perform the modulo operation again. If this yields the same address, then increment it once and perform the modulo yet again.

Of course once it is using an IP address, if other workstations attempt to assign that address, normal ARP replies will act to ‘‘defend’’ its address. During the window between deciding it is OK to use a

[1]‘‘Important’’ in as much as it is felt that a workstation should tend to be assigned the same IP address if at all possible across reboots.

particular address and starting to use it, another host may choose the same address. Therefore it is necessary to perform the ARP test again after setting your address. This window is particularly vulnerable after a power glitch when many workstations may run this algorithm at the same time. If the second ARP test fails, a new address must be chosen.

## Network Information Protocol

The network information protocol will operate directly on the Ethernet layer. It will use a Ethernet type value that is unique to it (it is not implemented as an extension of the ARP protocol, as correct functioning with all existing ARP implementations cannot be verified).

Note that this protocol uses some of the ideas of ‘‘Reverse ARP’’ (RFC 903), but is primarily different in the additional information returned, and that the client does not have to get back an address as an authoritative answer.

The first step is to prepare to recieve responses, then wait a short period of time. The recommended wait is 1 second plus the last byte of the physical ethernet address times 10 milliseconds. These times may be adjusted to fit the timer resolution of the supporting operating system. If during this time a sufficient response is received, then it is no longer necessary to send a request.

To start the protocol a workstation sends a broadcast packet which requests the information. Upon receiving such a request, all workstations will forward the information to the requesting workstation (hopefully this information is relatively constant). To prevent too many collisions and sudden network congestion, each workstation will wait a short amount of time before responding. The recommended wait is 100 milliseconds plus the last byte of the IP address times 1 millisecond. As above, these times may be adjusted to fit the timer resolution of the supporting operating system. At least one workstation/host on the network should be configured as a primary NIP server and will respond immediately. In general the ‘‘primary’’ information provider will be the default gateway (though it doesn’t have to be). The ‘‘primary’’ provider will also be the only information provider that will fill in the otherwise optional ‘‘Recommended IP address’’ field. It can obtain this information by using a ‘‘Reverse ARP’’ lookup on its ARP cache. The word ‘‘can’’ should be emphasized here, since although reverse ARP may be a clever way to help pick an IP address, it is not the only way that this can be done. This protocol specification does not preclude the use of another method.

A workstation which has not yet determined its IP address WILL NOT respond to an information request.

The information response packet will contain the following information. Addresses and 16 bit quantities are transmitted in network standard byte order, that is most significant byte first.

- A Version number, because version numbers a good and virtuous in that they allow future enhancement of the protocol in an upward compatible way!
- The opcode indicating a request or a reply.
- The source hardware address. This is repeated in the packet for ease of replying when

lower level software may have lost the physical header.

- A checksum, which is the compliment of the ones compliment sum of the 16 bit words in the packet. This is the same algorithm as the IP header, and is necessary for the protocol to be useful on other media and detecting malfunctioning hardware.
- Network address of the network. This will be an IP address with a host portion field of 0.
- A network mask. This will a 32bit quantity with ones masking off the network number portion of the IP address.
- A broadcast address. The broadcast address of choice on this network.
- The lowest available IP address for dynamic assignment.
- The highest available IP address for dynamic assignment.
- An optional recommended IP address for the requesting host to use. This field may be left zero.
- A list of addresses of IP gateways for this network.

```
+---------------------------------+
|Source Ethernet Address          |
+----------------+----------------+
|Source Ethernet |Checksum        |
+----------------+----------------+
|Opcode          |Version         | <-- Version = 1
+----------------+----------------+ <-- Below here opcode dependent
|Network Address                  |     These fields are only used
+---------------------------------+     when opcode = Response
|Network Mask                     |
+---------------------------------+
|Broadcast Address                |
+---------------------------------+
|Lowest usable IP address         |
+---------------------------------+
|Highest usable IP address        |
+---------------------------------+
|Recommended IP address           | <-- Either a first guess IP address
+---------------------------------+     to use, or zero
|Default Gateway address(es)      |
+---------------------------------+
|  More Gateway Addresses ...
+--------------//-----------------+
                                  |
+---------------------------------+
```

## Commentary

One of the important issues that an implementor of this protocol needs to be aware of, is how the ‘‘system’’ will react to sudden interruption of all workstations on the network (i.e. a power failure). Things should initialize correctly even if all devices on the network are suddenly reinitializing. In other words someone **has** to be the authority on the information provided in the ‘‘Network Information Protocol.’’ The authority presumably **knows** what this information is and doesn’t have to ask when he initializes.

Consider the following scenario. A network can be set up so that the default gateway is the ‘‘primary’’ (and authoritative) information provider. We assume that the primary **knows** its configuration through mechanisms outside of this protocol and thus **knows** its IP address on the ethernet and the other information provided by the ‘‘Network Information Protocol.’’

If all systems reboot simultaneously (as after a power outage) the following occurs: Workstations will broadcast ''Network Information Request'' packets. Until the primary comes up, no one will respond (because until a host knows its IP address it must not respond to Network Information Requests). When the primary comes up then it can start responding and the workstations come up. If Reverse ARP is normally used by the primary to provide the ''Recommended IP address'', this information can be left out (because obviously the primary's ARP cache is probably empty at this point).

Of course in this scenario the workstations will fail to initialize their network interfaces unless (or until) the primary comes up. This is probably a reasonable price to pay, for chances are if the primary is also the gateway that candidates for this protocol probably depend on other network services provided through that gateway (like RFC882 domain name resolution for example) and cannot get much utility out of the network until the gateway comes up anyway.

This protocol has not addressed finding a hostname to go with the address. Since this may require communication with domain name servers, it will involve yet another protocol. The obvious first choice is to reverse-resolve the chosen address to get a name. If no name has been assigned to that address, then the host and local domain name server must negotiate a name. If the host chooses a name that is not registered with the name servers, other sites will not be able to initiate connections to the workstation.

A final issue that has not been addressed to dynamically configure the network is the location of name, file, print, mail and other servers. Protocols exist for this, such as the Resource Location Protocol of RFC887 and various nameservers such as Hesiod or Yellowpages.

## References

1. Ross Finlayson, Timothy Mann, Jeffrey Mogul, Marvin Theimer. A Reverse Address Resolution Protocol. RFC 903, NIC, June, 1984.
2. David Plummer. An Ethernet Address Resolution Protocol. RFC 826, NIC, September, 1982.
3. Bill Croft, John Gilmore. Bootstrap Protocol. RFC 951, NIC, September 1985.
4. M. Accetta. Resource Location Protocol. RFC 887, NIC, December 1983.