Internet Draft: DHC-IPV4-AUTOCONFIG
Document: draft-ietf-dhc-ipv4-autoconfig-01.txt
Expires: April 1999

R. Troll
October 1998

Automaticly Choosing an IP Address in an Ad-Hoc IPv4 Network

<draft-ietf-dhc-ipv4-autoconfig-01.txt>

Status of this memo

Abstract

With operating systems appearing in more and more devices, as well as computers appearing in more and more aspects of everyday life, communication between networked devices is increasingly important. The communication mechanism between these devices must be able to not only support the office LAN environment, but must also scale to larger WANS and the internet.

This draft describes a method by which a host may automaticly give itself a link-local IPv4 address, so that it will be able to use IP applications in the absence of an IP address management mechanism, such as DHCP. This mechanism is in use today by a few operating systems, and additional information on those implementations is also provided.

1. Introduction

Now that networked applications are becoming more prevalent, operating systems are migrating towards more scalable network protocols such as IP, allowing them to work in all sizes of environments. However, there is a price to pay for this migration -- IP requires configuration that other protocols (IPX, Appletalk) do not require.

Dynamic creation of usable ad-hoc networks is very useful when there are only a few machines on the entire network. (For example, a dentist's office may only have a couple of machines.) In order to allow a site such as this to use IP, the machines must each be configured with an IP address. OS's wish to retain the minimal configuration that was necessary under their non-IP network stacks.

Dynamic configuration protocols such as DHCP [DHCP] allow a site administrator to take care of the network configuration for a machine remotely. By requesting network parameters via DHCP, the site administrator may provide all information necessary without the host's owner having to do anything. However, not all sites have a central administrator to take care of this.

To accommodate unmanaged networks, the OS may decide to intelligently choose an IP address for itself. These addresses are only valid for the local network.

This document describes a method by which an OS may determine whether or not to autoconfigure itself an IP address, as well as how to inter-operate cleanly with an existing managed infrastructure, allowing a host to easily move between managed and unmanaged network segments.

1.1 Conventions Used in the Document

The key words "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", and "MAY" in this document are to be interpreted as defined in "Key Words for Use in RFCs to Indicate Requirement Levels" [KEYWORDS]

1.2 Terminology

Site Administrator
: A Site Administrator is the person or organization responsible for handing out IP addresses to client machines.

DHCP Client
: A DHCP Client is an Internet host using DHCP to obtain configuration parameters such as a network address.

DHCP Server A DHCP Server is an Internet host that returns configuration parameters to DHCP Clients.

1.3 Usage Clarification

This document describes a method by which a host may automaticly choose an IPv4 address in the absence of a central service to maintain and hand out addresses. This is not designed to replace this functionality, but to basicly provide it in small networks.

This SHOULD not be used for large-scale networks. As more and more machines begin to use this mechanism on a network, startup times for these machines will begin to decrease, as the chance of collisions will rise.

Addresses allocated by this mechanism MUST NOT be routed by any network device. The addresses are designed to be link local addresses. Link local address are to be, by definition, restricted to the local network segment. Allocation of link-local addresses in an IPv6 network is described in [IPv6SAC].

2. To Choose or Not To Choose

The first thing an Internet host should do is request an IP address via DHCP [DHCP]. This is done by sending out a DHCPDISCOVER message, with various tags set indicating what options the DHCP Client would like to receive information for [DHCPOPT]. The DHCP Client SHOULD also send the DHCP AutoConfigure option described in [DHCPAC].

According to [DHCP], Section 4.4.1, the amount of time over which a DHCP Client should listen for DHCPOFFERS is implementation dependant. During this time, if a DHCPOFFER is received, network configuration MUST occur as described in [DHCP] and [DHCPAC].

If, during this time, no valid DHCPOFFERS are received, the DHCP Client is free to autoconfigure an IP address according to section 3 of this document.

2.1 Rebinding an Existing IP Address

If the DHCP Client already has an existing IP address, it MUST follow the instructions outlined in [DHCP]. If the client winds up back in the INIT state, refer to section 2 of this document.

3. Choosing an IP Address

Once a DHCP Client has determined it must auto-configure an IP

address, it chooses an address. The algorithm for choosing an address is implementation dependant. The address range to use MUST be "169.254/16", which is registered with the IANA as the LINKLOCAL net.

If choosing an address in this range, the DHCP Client MUST not use the first 256 or the last 256, as these are reserved for future use.

When an address is chosen, the DHCP Client MUST test to see if the address is already in use. If the network address appears to be in use, the client MUST choose another address, and try again. The client MUST keep choosing addresses until it either finds one, or it has tried more then the autoconfig-retry count. The autoconfig-retry count is implementation specific, and should be based on the algorithm used for choosing an IP address. This retry count is present to make sure that DHCP Clients auto-configuring on busy auto-configured network segments do not loop infinitely looking for an IP address.

3.1 Determining Whether or Not an Address is in Use

If the client is on a network that supports ARP, the client may issue an ARP request for the suggested address. When broadcasting an ARP request for the suggested address, the client MUST fill in its own hardware address as the sender's hardware address, and all 0s as the sender's IP address, to avoid confusing ARP caches in other hosts on the same subnet. This ARP request with a sender IP address of all 0s is referred to as an "ARP probe".

While waiting for a possible response to this request, the client MUST also listen for other ARP probes for the same address (but not from its own hardware address). This will occur if two (or more) hosts are attempting to autoconfigure the exact same address. If the client receives a response to the ARP request, or sees another ARP probe for the same address, it MUST consider the address as being in use, and move on.

4. Ongoing Checks for a DHCP Server

When the client originally sent out it's request, there may have been a network problem stopping the DHCP Server from responding. To make sure this is not the case, a DHCP Client with an auto-configured IP address MUST keep checking for an active DHCP Server. To do this, the DHCP Client MUST attempt to fetch an IP address as

described in section 1 of this document.

When rechecking, when the DHCP Client has determined no DHCP Server is responding, it MUST wait a period of time and try again. For Ethernet implementations, the DHCP Client SHOULD check every 5 minutes.

If the DHCP Client receives a response from a DHCP Server, it MUST respond and attempt to obtain a lease from the server (per the DHCP specification). If the client is successful in obtaining a new lease, and the internet host does not support multiple addresses on the interface being configured, it MUST drop any existing auto-configured IP address, and all active connections, while moving to the new address. If the internet host does support multiple addresses on the interface, it MAY keep the auto-configured address active.

If the DHCP response is an AutoConfigure [DHCPAC] response set to "DoNOTAutoConfigure", the host MUST drop all connections, give up any existing auto-configured IP address, and continue checking for a DHCP server.

5. Current Vendor Implementations

As of this writing, Microsoft and Apple have operating systems that contain this functionality. Descriptions of the implementation dependant parts are listed below.

5.1. Microsoft Windows 98

With the initial release of Windows 98, Microsoft introduced auto-configuration functionality. When developed, the AutoConfig [DHCPAC] specification did not exist, so the initial release does not contain this functionality.

The Win98 DHCP Client sends out a total of 4 DHCPDISCOVERs, with an inter-packet interval of 6 seconds. When no response is received after all 4 packets (24 seconds), it will auto-configure an address.

The auto-configure retry count for Windows 98 is 10. After trying 10 auto-configured IP addresses, and finding all are taken, the host will boot without an IP address.

5.2. Apple MacOS 8.5

MacOS 8.5 sends three DHCPDISCOVER packets, with timeouts of 4, 8, and then 16 seconds. When no response is received from all of these requests (28 seconds), it will auto-configure.

The auto-configure retry count for MacOS 8.5 is 10. After trying 10 auto-configured IP addresses, and finding all are taken, the host will boot without an IP address.

6. Security Considerations

The use of this functionality may open a network host to new Denial Of Service (DOS) attacks. In particular, a host that previously did not have an IP address, and no IP stack running, was not succeptable to IP based DOS attacks, as there was no IP stack configured to interpret these packets.

However, the addition of this functionality to an OS may cause IP stacks to be capable of receiving and interpreting information that the host was not previously configured to receive. As this how is now interpreting IP communications, it is now open to IP based DOS attacks.

Another security concern is the DOS attack that may be made on the local subnet which stops all machines from being able to allocate an IP address. A malicious host on the local wire may listen for ARP probes, and respond with it's own ARP probe. This will stop the auto-configuring machine from using that address, and it will move on to the next one. Eventually, it will run out of addresses to attempt, and will give up. The use of DHCP removes this attack, leaving only the concerns described in [DHCP].

Finally, machines that rely on this for communication over a large network may allocate the same address if the network itself is segmented when the machines boot. If the link between two machines is down when they boot, they may both auto-configure the same address. However, when the network link returns, there will be numerous problems (ARP caches, etc.) There is currently no way to solve this auto-configuration problem without causing all hosts involved to re-autoconfigure IP addresses. The use of DHCP to configure hosts on a subnet will solve this, and hosts that implement this configuration mechanism will behave appropriately on a DHCP managed network in which the DHCP server is not initially available.

7. Acknowledgments

I'd like to thank Microsoft and Apple for their help in writing

this document.

8. Copyright

9. References

[DHCP] Droms, R. "Dynamic Host Configuration Protocol", RFC 2131, Bucknell University, March 1997.

<ftp://ds.internic.net/rfc/rfc2131.txt>

[DHCPOPT] Alexander, S. and Droms, R., "DHCP Options and BOOTP Vendor Extension", RFC 2132, March 1997.

<ftp://ds.internic.net/rfc/rfc2132.txt>

[KEYWORDS] Bradner, "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, Harvard University, March 1997.

<ftp://ds.internic.net/rfc/rfc2119.txt>

[IPv6SAC] Thomson, S. and Narten, T. "IPv6 Stateless Address Autoconfiguration", RFC 1971, August 1996

<ftp://ds.internic.net/rfc/rfc1971.txt>

[DHCPAC] Troll, R. "DHCP Option to Disable Stateless Auto-Configuration in IPv4 Clients", RFC XXXXX, November 1998

<ftp://ds.internic.net/rfc/rfcXXXXX.txt>

10. Author's Address

Ryan Troll
Network Development
Carnegie Mellon
5000 Forbes Avenue
Pittsburgh, PA 15213

Phone: (412) 268-8691
EMail: ryan@andrew.cmu.edu

This document will expire April 1999