Network Working Group
Request for Comments: 5227
Updates: 826
Category: Standards Track

S. Cheshire
Apple Inc.
July 2008

# IPv4 Address Conflict Detection

## Status of This Memo

## Abstract

When two hosts on the same link attempt to use the same IPv4 address at the same time (except in rare special cases where this has been arranged by prior coordination), problems ensue for one or both hosts. This document describes (i) a simple precaution that a host can take in advance to help prevent this misconfiguration from happening, and (ii) if this misconfiguration does occur, a simple mechanism by which a host can passively detect, after the fact, that it has happened, so that the host or administrator may respond to rectify the problem.

Table of Contents

1. Introduction ....................................................2
   1.1. Conventions and Terminology Used in This Document ..........4
   1.2. Relationship to RFC 826 ....................................5
        1.2.1. Broadcast ARP Replies ...............................7
   1.3. Applicability ..............................................7
2. Address Probing, Announcing, Conflict Detection, and Defense ....9
   2.1. Probing an Address ........................................10
        2.1.1. Probe Details ......................................10
   2.2. Shorter Timeouts on Appropriate Network Technologies ......11
   2.3. Announcing an Address .....................................12
   2.4. Ongoing Address Conflict Detection and Address Defense ....12
   2.5. Continuing Operation ......................................14
   2.6. Broadcast ARP Replies .....................................14
3. Why Are ARP Announcements Performed Using ARP Request
   Packets and Not ARP Reply Packets? .............................15
4. Historical Note ................................................17
5. Security Considerations ........................................17
6. Acknowledgments ................................................18
7. References .....................................................18
   7.1. Normative References ......................................18
   7.2. Informative References ....................................19

## 1. Introduction

Historically, accidentally configuring two Internet hosts with the same IP address has often been an annoying and hard-to-diagnose problem.

This is unfortunate, because the existing Address Resolution Protocol (ARP) provides an easy way for a host to detect this kind of misconfiguration and report it to the user. The DHCP specification [RFC2131] briefly mentions the role of ARP in detecting misconfiguration, as illustrated in the following three excerpts from RFC 2131:

- o the client SHOULD probe the newly received address, e.g., with ARP
- o The client SHOULD perform a final check on the parameters (e.g., ARP for allocated network address)
- o If the client detects that the address is already in use (e.g., through the use of ARP), the client MUST send a DHCPDECLINE message to the server

Unfortunately, the DHCP specification does not give any guidance to implementers concerning the number of ARP packets to send, the interval between packets, the total time to wait before concluding that an address may safely be used, or indeed even which kinds of packets a host should be listening for, in order to make this determination. It leaves unspecified the action a host should take if, after concluding that an address may safely be used, it subsequently discovers that it was wrong. It also fails to specify what precautions a DHCP client should take to guard against pathological failure cases, such as a DHCP server that repeatedly OFFERs the same address, even though it has been DECLINEd multiple times.

The authors of the DHCP specification may have been justified in thinking at the time that the answers to these questions seemed too simple, obvious, and straightforward to be worth mentioning, but unfortunately this left some of the burden of protocol design to each individual implementer. This document seeks to remedy this omission by clearly specifying the required actions for:

1. Determining whether use of an address is likely to lead to an addressing conflict. This includes (a) the case where the address is already actively in use by another host on the same link, and (b) the case where two hosts are inadvertently about to begin using the same address, and both are simultaneously in the process of probing to determine whether the address may safely be used (Section 2.1.).

2. Subsequent passive detection that another host on the network is inadvertently using the same address. Even if all hosts observe precautions to avoid using an address that is already in use, conflicts can still occur if two hosts are out of communication at the time of initial interface configuration. This could occur with wireless network interfaces if the hosts are temporarily out of range, or with Ethernet interfaces if the link between two Ethernet hubs is not functioning at the time of address configuration. A well-designed host will handle not only conflicts detected during interface configuration, but also conflicts detected later, for the entire duration of the time that the host is using the address (Section 2.4.).

3. Rate-limiting of address acquisition attempts in the case of an excessive number of repeated conflicts (Section 2.1.).

The utility of IPv4 Address Conflict Detection (ACD) is not limited to DHCP clients. No matter how an address was configured, whether via manual entry by a human user, via information received from a DHCP server, or via any other source of configuration information, detecting conflicts is useful. Upon detecting a conflict, the configuring agent should be notified of the error. In the case where the configuring agent is a human user, that notification may take the form of an error message on a screen, a Simple Network Management Protocol (SNMP) notification, or an error message sent via text message to a mobile phone. In the case of a DHCP server, that notification takes the form of a DHCP DECLINE message sent to the server. In the case of configuration by some other kind of software, that notification takes the form of an error indication to the software in question, to inform it that the address it selected is in conflict with some other host on the network. The configuring software may choose to cease network operation, or it may automatically select a new address so that the host may re-establish IP connectivity as soon as possible.

Allocation of IPv4 Link-Local Addresses [RFC3927] can be thought of as a special case of this mechanism, where the configuring agent is a pseudo-random number generator, and the action it takes upon being notified of a conflict is to pick a different random number and try again. In fact, this is exactly how IPv4 Link-Local Addressing was implemented in Mac OS 9 back in 1998. If the DHCP client failed to get a response from any DHCP server, it would simply make up a fake response containing a random 169.254.x.x address. If the ARP module reported a conflict for that address, then the DHCP client would try again, making up a new random 169.254.x.x address as many times as was necessary until it succeeded. Implementing ACD as a standard feature of the networking stack has the side effect that it means that half the work for IPv4 Link-Local Addressing is already done.

## 1.1. Conventions and Terminology Used in This Document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119].

Wherever this document uses the term 'sender IP address' or 'target IP address' in the context of an ARP packet, it is referring to the fields of the ARP packet identified in the ARP specification [RFC826] as 'ar$spa' (Sender Protocol Address) and 'ar$tpa' (Target Protocol Address), respectively. For the usage of ARP described in this document, each of these fields always contains an IPv4 address.

In this document, the term 'ARP Probe' is used to refer to an ARP Request packet, broadcast on the local link, with an all-zero 'sender IP address'. The 'sender hardware address' MUST contain the hardware address of the interface sending the packet. The 'sender IP address' field MUST be set to all zeroes, to avoid polluting ARP caches in other hosts on the same link in the case where the address turns out to be already in use by another host. The ’target hardware address’ field is ignored and SHOULD be set to all zeroes. The ’target IP address’ field MUST be set to the address being probed. An ARP Probe conveys both a question ("Is anyone using this address?") and an implied statement ("This is the address I hope to use.").

In this document, the term ’ARP Announcement’ is used to refer to an ARP Request packet, broadcast on the local link, identical to the ARP Probe described above, except that both the sender and target IP address fields contain the IP address being announced. It conveys a stronger statement than an ARP Probe, namely, "This is the address I am now using."

The following timing constants used in this protocol are referenced in Section 2, which describes the operation of the protocol in detail. (Note that the values listed here are fixed constants; they are not intended to be modifiable by implementers, operators, or end users. These constants are given symbolic names here to facilitate the writing of future standards that may want to reference this document with different values for these named constants; however, at the present time no such future standards exist.)

| | | |
|---|---|---|
| PROBE_WAIT | 1 second | (initial random delay) |
| PROBE_NUM | 3 | (number of probe packets) |
| PROBE_MIN | 1 second | (minimum delay until repeated probe) |
| PROBE_MAX | 2 seconds | (maximum delay until repeated probe) |
| ANNOUNCE_WAIT | 2 seconds | (delay before announcing) |
| ANNOUNCE_NUM | 2 | (number of Announcement packets) |
| ANNOUNCE_INTERVAL | 2 seconds | (time between Announcement packets) |
| MAX_CONFLICTS | 10 | (max conflicts before rate-limiting) |
| RATE_LIMIT_INTERVAL | 60 seconds | (delay between successive attempts) |
| DEFEND_INTERVAL | 10 seconds | (minimum interval between defensive ARPs) |

## 1.2. Relationship to RFC 826

This document does not modify any of the protocol rules in RFC 826. It does not modify the packet format, or the meaning of any of the fields. The existing rules for "Packet Generation" and "Packet Reception" still apply exactly as specified in RFC 826.

This document expands on RFC 826 by specifying:

(1) that a specific ARP Request should be generated when an interface is configured, to discover if the address is already in use, and

(2) an additional trivial test that should be performed on each received ARP packet, to facilitate passive ongoing conflict detection. This additional test creates no additional packet overhead on the network (no additional packets are sent) and negligible additional CPU burden on hosts, since every host implementing ARP is *already* required to process every received ARP packet according to the Packet Reception rules specified in RFC 826. These rules already include checking to see if the ’sender IP address’ of the ARP packet appears in any of the entries in the host’s ARP cache; the additional test is simply to check to see if the ’sender IP address’ is the host’s *own* IP address, potentially as little as a single additional machine instruction on many architectures.

As already specified in RFC 826, an ARP Request packet serves two functions, an assertion and a question:

* Assertion:
  The fields ’ar$sha’ (Sender Hardware Address) and ’ar$spa’ (Sender Protocol Address) together serve as an assertion of a fact: that the stated Protocol Address is mapped to the stated Hardware Address.

* Question:
  The fields ’ar$tha’ (Target Hardware Address, zero) and ’ar$tpa’ (Target Protocol Address) serve as a question, asking, for the stated Protocol Address, to which Hardware Address it is mapped.

This document clarifies what it means to have one without the other.

Some readers pointed out that it is probably impossible to ask any truly pure question; asking any question necessarily invites speculation about why the interrogator wants to know the answer. Just as someone pointing to an empty seat and asking, "Is anyone sitting here?" implies an unspoken "... because if not then I will," the same is true here. An ARP Probe with an all-zero ’sender IP address’ may ostensibly be merely asking an innocent question ("Is anyone using this address?"), but an intelligent implementation that knows how IPv4 Address Conflict Detection works should be able to recognize this question as the precursor to claiming the address.

Consequently, if that implementation is also, at that exact moment, in the process of asking the very same question, it should recognize that they can’t both sit in the same seat, so it would be prudent to ask about some other seat.

### 1.2.1. Broadcast ARP Replies

In some applications of IPv4 Address Conflict Detection (ACD), it may be advantageous to deliver ARP Replies using broadcast instead of unicast because this allows address conflicts to be detected sooner than might otherwise happen. For example, "Dynamic Configuration of IPv4 Link-Local Addresses" [RFC3927] uses ACD exactly as specified here, but additionally specifies that ARP Replies should be sent using broadcast, because in that context the trade-off of increased broadcast traffic in exchange for improved reliability and fault-tolerance was deemed to be an appropriate one. There may be other future specifications where the same trade-off is appropriate. Additional details are given in Section 2.6, "Broadcast ARP Replies".

RFC 826 implies that replies to ARP Requests are usually delivered using unicast, but it is also acceptable to deliver ARP Replies using broadcast. The Packet Reception rules in RFC 826 specify that the content of the 'ar$spa' field should be processed *before* examining the 'ar$op' field, so any host that correctly implements the Packet Reception algorithm specified in RFC 826 will correctly handle ARP Replies delivered via link-layer broadcast.

## 1.3. Applicability

This specification applies to all IEEE 802 Local Area Networks (LANs) [802], including Ethernet [802.3], Token-Ring [802.5], and IEEE 802.11 wireless LANs [802.11], as well as to other link-layer technologies that operate at data rates of at least 1 Mb/s, have a round-trip latency of at most one second, and use ARP [RFC826] to map from IP addresses to link-layer hardware addresses. Wherever this document uses the term "IEEE 802", the text applies equally to any of these network technologies.

Link-layer technologies that support ARP but operate at rates below 1 Mb/s or latencies above one second will still work correctly with this protocol, but more often may have to handle late conflicts detected after the Probing phase has completed. On these kinds of links, it may be desirable to specify different values for the following parameters:

(a) PROBE_NUM, PROBE_MIN, and PROBE_MAX, the number of, and interval between, ARP Probes, explained in Section 2.1.

(b) ANNOUNCE_NUM and ANNOUNCE_INTERVAL, the number of, and interval between, ARP Announcements, explained in Section 2.3.

(c) RATE_LIMIT_INTERVAL and MAX_CONFLICTS, controlling the maximum rate at which address claiming may be attempted, explained in Section 2.1.

(d) DEFEND_INTERVAL, the time interval between conflicting ARPs below which a host MUST NOT attempt to defend its address, explained in Section 2.4.

Link-layer technologies that do not support ARP may be able to use other techniques for determining whether a particular IP address is currently in use. However, implementing Address Conflict Detection for such networks is outside the scope of this document.

For the protocol specified in this document to be effective, it is not necessary that all hosts on the link implement it. For a given host implementing this specification to be protected against accidental address conflicts, all that is required is that the peers on the same link correctly implement the ARP protocol as given in RFC 826. To be specific, when a peer host receives an ARP Request where the Target Protocol Address of the ARP Request matches (one of) that host’s IP address(es) configured on that interface, then as long as it properly responds with a correctly-formatted ARP Reply, the querying host will be able to detect that the address is already in use.

The specifications in this document allow hosts to detect conflicts between two hosts using the same address on the same physical link. ACD does not detect conflicts between two hosts using the same address on different physical links, and indeed it should not. For example, the address 10.0.0.1 [RFC1918] is in use by countless devices on countless private networks throughout the world, and this is not a conflict, because they are on different links. It would only be a conflict if two such devices were to be connected to the same link, and when this happens (as it sometimes does), this is a perfect example of a situation where ACD is extremely useful to detect and report (and/or automatically correct) this error.

For the purposes of this document, a set of hosts is considered to be "on the same link" if:

- when any host, A, from that set, sends a packet to any other host, B, in that set, using unicast, multicast, or broadcast, the entire link-layer packet payload arrives unmodified, and

- a broadcast sent over that link by any host from that set of hosts can be received by every other host in that set.

The link-layer *header* may be modified, such as in Token Ring Source Routing [802.5], but not the link-layer *payload*. In particular, if any device forwarding a packet modifies any part of the IP header or IP payload, then the packet is no longer considered to be on the same link. This means that the packet may pass through devices such as repeaters, bridges, hubs, or switches and still be considered to be on the same link for the purpose of this document, but not through a device such as an IP router that decrements the TTL or otherwise modifies the IP header.

Where this document uses the term "host", it applies equally to interfaces on routers or other multi-homed hosts, regardless of whether the host/router is currently forwarding packets. In many cases a router will be critical network infrastructure with an IP address that is locally well known and assumed to be relatively constant. For example, the address of the default router is one of the parameters that a DHCP server typically communicates to its clients, and (at least until mechanisms like DHCP Reconfigure [RFC3203] become widely implemented) there isn’t any practical way for the DHCP server to inform clients if that address changes. Consequently, for such devices, handling conflicts by picking a new IP address is not a good option. In those cases, option (c) in Section 2.4 ("Ongoing Address Conflict Detection and Address Defense") applies.

However, even when a device is manually configured with a fixed address, having some other device on the network claiming to have the same IP address will pollute peer ARP caches and prevent reliable communication, so it is still helpful to inform the operator. If a conflict is detected at the time the operator sets the fixed manual address, then it is helpful to inform the operator immediately; if a conflict is detected subsequently, it is helpful to inform the operator via some appropriate asynchronous communication channel. Even though reliable communication via the conflicted address is not possible, it may still be possible to inform the operator via some other communication channel that is still operating, such as via some other interface on the router, via a dynamic IPv4 link-local address, via a working IPv6 address, or even via some completely different non-IP technology such as a locally-attached screen or serial console.

## 2. Address Probing, Announcing, Conflict Detection, and Defense

This section describes initial probing to safely determine whether an address is already in use, announcing the chosen address, ongoing conflict checking, and optional use of broadcast ARP Replies to provide faster conflict detection.

## 2.1. Probing an Address

Before beginning to use an IPv4 address (whether received from manual configuration, DHCP, or some other means), a host implementing this specification MUST test to see if the address is already in use, by broadcasting ARP Probe packets. This also applies when a network interface transitions from an inactive to an active state, when a computer awakes from sleep, when a link-state change signals that an Ethernet cable has been connected, when an 802.11 wireless interface associates with a new base station, or when any other change in connectivity occurs where a host becomes actively connected to a logical link.

A host MUST NOT perform this check periodically as a matter of course. This would be a waste of network bandwidth, and is unnecessary due to the ability of hosts to passively discover conflicts, as described in Section 2.4.

### 2.1.1. Probe Details

A host probes to see if an address is already in use by broadcasting an ARP Request for the desired address. The client MUST fill in the 'sender hardware address' field of the ARP Request with the hardware address of the interface through which it is sending the packet. The 'sender IP address' field MUST be set to all zeroes; this is to avoid polluting ARP caches in other hosts on the same link in the case where the address turns out to be already in use by another host. The 'target hardware address' field is ignored and SHOULD be set to all zeroes. The 'target IP address' field MUST be set to the address being probed. An ARP Request constructed this way, with an all-zero 'sender IP address', is referred to as an 'ARP Probe'.

When ready to begin probing, the host should then wait for a random time interval selected uniformly in the range zero to PROBE_WAIT seconds, and should then send PROBE_NUM probe packets, each of these probe packets spaced randomly and uniformly, PROBE_MIN to PROBE_MAX seconds apart. This initial random delay helps ensure that a large number of hosts powered on at the same time do not all send their initial probe packets simultaneously.

If during this period, from the beginning of the probing process until ANNOUNCE_WAIT seconds after the last probe packet is sent, the host receives any ARP packet (Request *or* Reply) on the interface where the probe is being performed, where the packet's 'sender IP address' is the address being probed for, then the host MUST treat this address as being in use by some other host, and should indicate to the configuring agent (human operator, DHCP server, etc.) that the proposed address is not acceptable.

In addition, if during this period the host receives any ARP Probe where the packet's 'target IP address' is the address being probed for, and the packet's 'sender hardware address' is not the hardware address of any of the host's interfaces, then the host SHOULD similarly treat this as an address conflict and signal an error to the configuring agent as above. This can occur if two (or more) hosts have, for whatever reason, been inadvertently configured with the same address, and both are simultaneously in the process of probing that address to see if it can safely be used.

NOTE: The check that the packet's 'sender hardware address' is not the hardware address of any of the host's interfaces is important. Some kinds of Ethernet hub (often called a "buffered repeater") and many wireless access points may "rebroadcast" any received broadcast packets to all recipients, including the original sender itself. For this reason, the precaution described above is necessary to ensure that a host is not confused when it sees its own ARP packets echoed back.

A host implementing this specification MUST take precautions to limit the rate at which it probes for new candidate addresses: if the host experiences MAX_CONFLICTS or more address conflicts on a given interface, then the host MUST limit the rate at which it probes for new addresses on this interface to no more than one attempted new address per RATE_LIMIT_INTERVAL. This is to prevent catastrophic ARP storms in pathological failure cases, such as a defective DHCP server that repeatedly assigns the same address to every host that asks for one. This rate-limiting rule applies not only to conflicts experienced during the initial probing phase, but also to conflicts experienced later, as described in Section 2.4 "Ongoing Address Conflict Detection and Address Defense".

If, by ANNOUNCE_WAIT seconds after the transmission of the last ARP Probe no conflicting ARP Reply or ARP Probe has been received, then the host has successfully determined that the desired address may be used safely.

## 2.2. Shorter Timeouts on Appropriate Network Technologies

Network technologies may emerge for which shorter delays are appropriate than those required by this document. A subsequent IETF publication may be produced providing guidelines for different values for PROBE_WAIT, PROBE_NUM, PROBE_MIN, and PROBE_MAX on those technologies.

If the situation arises where different hosts on a link are using different timing parameters, this does not cause any problems. This protocol is not dependent on all hosts on a link implementing the same version of the protocol; indeed, this protocol is not dependent on all hosts on a link implementing the protocol at all. All that is required is that all hosts implement ARP as specified in RFC 826, and correctly answer ARP Requests they receive. In the situation where different hosts are using different timing parameters, all that will happen is that some hosts will configure their interfaces more quickly than others. In the unlikely event that an address conflict is not detected during the address probing phase, the conflict will still be detected by the Ongoing Address Conflict Detection described below in Section 2.4.

## 2.3. Announcing an Address

Having probed to determine that a desired address may be used safely, a host implementing this specification MUST then announce that it is commencing to use this address by broadcasting ANNOUNCE_NUM ARP Announcements, spaced ANNOUNCE_INTERVAL seconds apart. An ARP Announcement is identical to the ARP Probe described above, except that now the sender and target IP addresses are both set to the host’s newly selected IPv4 address. The purpose of these ARP Announcements is to make sure that other hosts on the link do not have stale ARP cache entries left over from some other host that may previously have been using the same address. The host may begin legitimately using the IP address immediately after sending the first of the two ARP Announcements; the sending of the second ARP Announcement may be completed asynchronously, concurrent with other networking operations the host may wish to perform.

## 2.4. Ongoing Address Conflict Detection and Address Defense

Address Conflict Detection is not limited to only the time of initial interface configuration, when a host is sending ARP Probes. Address Conflict Detection is an ongoing process that is in effect for as long as a host is using an address. At any time, if a host receives an ARP packet (Request *or* Reply) where the ’sender IP address’ is (one of) the host’s own IP address(es) configured on that interface, but the ’sender hardware address’ does not match any of the host’s own interface addresses, then this is a conflicting ARP packet, indicating some other host also thinks it is validly using this address. To resolve the address conflict, a host MUST respond to a conflicting ARP packet as described in either (a), (b), or (c) below:

(a) Upon receiving a conflicting ARP packet, a host MAY elect to immediately cease using the address, and signal an error to the configuring agent as described above.

(b) If a host currently has active TCP connections or other reasons to prefer to keep the same IPv4 address, and it has not seen any other conflicting ARP packets within the last DEFEND_INTERVAL seconds, then it MAY elect to attempt to defend its address by recording the time that the conflicting ARP packet was received, and then broadcasting one single ARP Announcement, giving its own IP and hardware addresses as the sender addresses of the ARP, with the ’target IP address’ set to its own IP address, and the ’target hardware address’ set to all zeroes. Having done this, the host can then continue to use the address normally without any further special action. However, if this is not the first conflicting ARP packet the host has seen, and the time recorded for the previous conflicting ARP packet is recent, within DEFEND_INTERVAL seconds, then the host MUST immediately cease using this address and signal an error to the configuring agent as described above. This is necessary to ensure that two hosts do not get stuck in an endless loop with both hosts trying to defend the same address.

(c) If a host has been configured such that it should not give up its address under any circumstances (perhaps because it is the kind of device that needs to have a well-known stable IP address, such as a link’s default router or a DNS server) then it MAY elect to defend its address indefinitely. If such a host receives a conflicting ARP packet, then it should take appropriate steps to log useful information such as source Ethernet address from the ARP packet, and inform an administrator of the problem. The number of such notifications should be appropriately controlled to prevent an excessive number of error reports being generated. If the host has not seen any other conflicting ARP packets recently, within the last DEFEND_INTERVAL seconds, then it MUST record the time that the conflicting ARP packet was received, and then broadcast one single ARP Announcement, giving its own IP and hardware addresses. Having done this, the host can then continue to use the address normally without any further special action. However, if this is not the first conflicting ARP packet the host has seen, and the time recorded for the previous conflicting ARP packet is within DEFEND_INTERVAL seconds, then the host MUST NOT send another defensive ARP Announcement. This is necessary to ensure that two misconfigured hosts do not get stuck in an endless loop flooding the network with broadcast traffic while they both try to defend the same address.

A host wishing to provide reliable network operation MUST respond to conflicting ARP packets as described in (a), (b), or (c) above. Ignoring conflicting ARP packets results in seemingly random network failures that can be hard to diagnose and very frustrating for human users.

Forced address reconfiguration may be disruptive, causing TCP (and other transport-layer) connections to be broken. However, such disruptions should be exceedingly rare, and if inadvertent address duplication happens, then disruption of communication is inevitable. It is not possible for two different hosts using the same IP address on the same network to operate reliably.

Before abandoning an address due to a conflict, hosts SHOULD actively attempt to reset any existing connections using that address. This mitigates some security threats posed by address reconfiguration, as discussed in Section 5.

For most client machines that do not need a fixed IP address, immediately requesting the configuring agent (human user, DHCP client, etc.) to configure a new address as soon as the conflict is detected is the best way to restore useful communication as quickly as possible. The mechanism described above of broadcasting a single ARP Announcement to defend the address mitigates the problem somewhat, by helping to improve the chance that one of the two conflicting hosts may be able to retain its address.

## 2.5. Continuing Operation

From the time a host sends its first ARP Announcement, until the time it ceases using that IP address, the host MUST answer ARP Requests in the usual way required by the ARP specification [RFC826]. Specifically, this means that whenever a host receives an ARP Request, that’s not a conflicting ARP packet as described above in Section 2.4, where the ’target IP address’ of the ARP Request is (one of) the host’s own IP address(es) configured on that interface, the host MUST respond with an ARP Reply as described in RFC 826. This applies equally for both standard ARP Requests with non-zero sender IP addresses and Probe Requests with all-zero sender IP addresses.

## 2.6. Broadcast ARP Replies

In a carefully-run network with manually-assigned addresses, or a network with a reliable DHCP server and reliable DHCP clients, address conflicts should occur only in rare failure scenarios, so the passive monitoring described above in Section 2.4 is adequate. If two hosts are using the same IP address, then sooner or later one host or the other will broadcast an ARP Request, which the other will see, allowing the conflict to be detected and consequently resolved.

It is possible, however, that a conflicting configuration may persist for a short time before it is detected. Suppose that two hosts, A and B, have been inadvertently assigned the same IP address, X. Suppose further that at the time they were both probing to determine whether the address could safely be used, the communication link between them was non-functional for some reason, so neither detected the conflict at interface-configuration time. Suppose now that the communication link is restored, and a third host, C, broadcasts an ARP Request for address X. Unaware of any conflict, both hosts A and B will send unicast ARP Replies to host C. Host C will see both Replies, and may be a little confused, but neither host A nor B will see the other's Reply, and neither will immediately detect that there is a conflict to be resolved. Hosts A and B will continue to be unaware of the conflict until one or other broadcasts an ARP Request of their own.

If quicker conflict detection is desired, this may be achieved by having hosts send ARP Replies using link-level broadcast, instead of sending only ARP Requests via broadcast, and Replies via unicast. This is NOT RECOMMENDED for general use, but other specifications building on IPv4 ACD may choose to specify broadcast ARP Replies if appropriate. For example, "Dynamic Configuration of IPv4 Link-Local Addresses" [RFC3927] specifies broadcast ARP Replies because in that context, detection of address conflicts using IPv4 ACD is not merely a backup precaution to detect failures of some other configuration mechanism; detection of address conflicts using IPv4 ACD is the sole configuration mechanism.

Sending ARP Replies using broadcast does increase broadcast traffic, but in the worst case by no more than a factor of two. In the traditional usage of ARP, a unicast ARP Reply only occurs in response to a broadcast ARP Request, so sending these via broadcast instead means that we generate at most one broadcast Reply in response to each existing broadcast Request. On many networks, ARP traffic is such an insignificant proportion of the total traffic that doubling it makes no practical difference. However, this may not be true of all networks, so broadcast ARP Replies SHOULD NOT be used universally. Broadcast ARP Replies should be used where the benefit of faster conflict detection outweighs the cost of increased broadcast traffic and increased packet processing load on the participant network hosts.

## 3. Why Are ARP Announcements Performed Using ARP Request Packets and Not ARP Reply Packets?

During IETF deliberation of IPv4 Address Conflict Detection from 2000 to 2008, a question that was asked repeatedly was, "Shouldn't ARP Announcements be performed using gratuitous ARP Reply packets?"

On the face of it, this seems reasonable. A conventional ARP Reply is an answer to a question. If in fact no question had been asked, then it would be reasonable to describe such a reply as gratuitous.

The term "gratuitous reply" would seem to apply perfectly to an ARP Announcement: an answer to an implied question that in fact no one asked.

However reasonable this may seem in principle, in practice there are two reasons that swing the argument in favor of using ARP Request packets. One is historical precedent, and the other is pragmatism.

The historical precedent is that (as described above in Section 4) Gratuitous ARP is documented in Stevens Networking [Ste94] as using ARP Request packets. BSD Unix, Microsoft Windows, Mac OS 9, Mac OS X, etc., all use ARP Request packets as described in Stevens. At this stage, trying to mandate that they all switch to using ARP Reply packets would be futile.

The practical reason is that ARP Request packets are more likely to work correctly with more existing ARP implementations, some of which may not implement RFC 826 entirely correctly. The Packet Reception rules in RFC 826 state that the opcode is the last thing to check in packet processing, so it really shouldn’t matter, but there may be "creative" implementations that have different packet processing depending on the ’ar$op’ field, and there are several reasons why these are more likely to accept gratuitous ARP Requests than gratuitous ARP Replies:

* An incorrect ARP implementation may expect that ARP Replies are only sent via unicast. RFC 826 does not say this, but an incorrect implementation may assume it; the "principle of least surprise" dictates that where there are two or more ways to solve a networking problem that are otherwise equally good, the one with the fewest unusual properties is the one likely to have the fewest interoperability problems with existing implementations. An ARP Announcement needs to broadcast information to all hosts on the link. Since ARP Request packets are always broadcast, and ARP Reply packets are not, receiving an ARP Request packet via broadcast is less surprising than receiving an ARP Reply packet via broadcast.

* An incorrect ARP implementation may expect that ARP Replies are only received in response to ARP Requests that have been issued recently by that implementation. Unexpected unsolicited Replies may be ignored.

* An incorrect ARP implementation may ignore ARP Replies where ’ar$tha’ doesn’t match its hardware address.

* An incorrect ARP implementation may ignore ARP Replies where ’ar$tpa’ doesn’t match its IP address.

In summary, there are more ways that an incorrect ARP implementation might plausibly reject an ARP Reply (which usually occurs as a result of being solicited by the client) than an ARP Request (which is already expected to occur unsolicited).

## 4. Historical Note

Some readers have claimed that "Gratuitous ARP", as described in Stevens [Ste94], provides duplicate address detection, making ACD unnecessary. This is incorrect. What Stevens describes as Gratuitous ARP is the exact same packet that this document refers to by the more descriptive term 'ARP Announcement'. This traditional Gratuitous ARP implementation sends only a single ARP Announcement when an interface is first configured. The result is that the victim (the existing address holder) logs an error, and the offender continues operation, often without even detecting any problem. Both machines then typically proceed to try to use the same IP address, and fail to operate properly because they are each constantly resetting the other's TCP connections. The human administrator is expected to notice the log message on the victim machine and repair the damage after the fact. Typically this has to be done by physically going to the machines in question, since in this state neither is able to keep a TCP connection open for long enough to do anything useful over the network.

Gratuitous ARP does not in fact provide effective duplicate address detection and (as of January 2008) many of the top results for a Google search for the phrase "Gratuitous ARP" are articles describing how to disable it.

However, implementers of IPv4 Address Conflict Detection should be aware that, as of this writing, Gratuitous ARP is still widely deployed. The steps described in Sections 2.1 and 2.4 of this document help make a host robust against misconfiguration and address conflicts, even when the other host is *not* playing by the same rules.

## 5. Security Considerations

IPv4 Address Conflict Detection (ACD) is based on ARP [RFC826] and it inherits the security vulnerabilities of that protocol. A malicious host may send fraudulent ARP packets on the network, interfering with the correct operation of other hosts. For example, it is easy for a host to answer all ARP Requests with Replies giving its own hardware address, thereby claiming ownership of every address on the network.

This specification makes this existing ARP vulnerability no worse, and in some ways makes it better: instead of failing silently with no indication why, hosts implementing this specification either attempt to reconfigure automatically, or at least inform the human user of what is happening.

If a host willingly selects a new address in response to an ARP conflict, as described in Section 2.4, subsection (a), this potentially makes it easier for malicious attackers on the same link to hijack TCP connections. Having a host actively reset any existing connections before abandoning an address helps mitigate this risk.

## 6. Acknowledgments

This document arose as a result of Zeroconf Working Group discussions on IPv4 Link-Local Addressing [RFC3927], where it was not clear to many participants which elements of link-local address management were specific to that particular problem space (e.g., random selection of an address), and which elements were generic and applicable to all IPv4 address configuration mechanisms (e.g., the detection of address conflicts). The following people made valuable comments in the course of that work and/or the subsequent editing of this document: Bernard Aboba, Randy Bush, Jim Busse, James Carlson, Alan Cox, Spencer Dawkins, Pavani Diwanji, Ralph Droms, Donald Eastlake III, Alex Elder, Stephen Farrell, Peter Ford, Spencer Giacalone, Josh Graessley, Erik Guttman, Myron Hattig, Mike Heard, Hugh Holbrook, Richard Johnson, Kim Yong-Woon, Marc Krochmal, Rod Lopez, Rory McGuire, Satish Mundra, Thomas Narten, Erik Nordmark, Randy Presuhn, Howard Ridenour, Pekka Savola, Daniel Senie, Dieter Siegmund, Valery Smyslov, Mark Townsley, Oleg Tychev, and Ryan Troll.

## 7. References

### 7.1. Normative References

[RFC826] Plummer, D., "An Ethernet Address Resolution Protocol -- or -- Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware", STD 37, RFC 826, November 1982.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## 7.2. Informative References

[802] IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture, ANSI/IEEE Std 802, 1990.

[802.3] ISO/IEC 8802-3 Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Common specifications - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications, (also ANSI/IEEE Std 802.3-1996), 1996.

[802.5] ISO/IEC 8802-5 Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Common specifications - Part 5: Token ring access method and physical layer specifications, (also ANSI/IEEE Std 802.5-1998), 1998.

[802.11] Information technology - Telecommunications and information exchange between systems - Local and metropolitan area networks - Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std. 802.11-1999, 1999.

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[RFC2131] Droms, R., "Dynamic Host Configuration Protocol", RFC 2131, March 1997.

[RFC3203] T'Joens, Y., Hublet, C., and P. De Schrijver, "DHCP reconfigure extension", RFC 3203, December 2001.

[RFC3927] Cheshire, S., Aboba, B., and E. Guttman, "Dynamic Configuration of IPv4 Link-Local Addresses", RFC 3927, May 2005.

[Ste94] W. Stevens, "TCP/IP Illustrated, Volume 1: The Protocols", Addison-Wesley, 1994.

## Author's Address

Stuart Cheshire
Apple Inc.
1 Infinite Loop
Cupertino
California 95014
USA

Phone: +1 408 974 3207
EMail: rfc@stuartcheshire.org