Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 8, 2011

D. Thaler
IAB

March 7, 2011

# Evolution of the IP Model
## draft-iab-ip-model-evolution-04.txt

Abstract

This draft attempts to document various aspects of the IP service model and how it has evolved over time. In particular, it attempts to document the properties of the IP layer as they are seen by upper-layer protocols and applications, and especially properties that were (and at times still are) incorrectly perceived to exist, as well as properties that would cause problems if changed. The discussion of these properties is organized around evaluating a set of claims, or misconceptions. Finally, this document provides some guidance to protocol designers and implementers.

Table of Contents

## 1. Introduction

Since the Internet Protocol was first published as [IEN028] in 1978, IP has provided a network-layer connectivity service to upper-layer protocols and applications. The basic IP service model was documented in the original IEN's (and subsequently in the RFC's that obsolete them). However, since the mantra has been "Everything Over IP", the IP service model has evolved significantly over the past 30 years to enable new behaviors that the original definition did not envision. For example, by 1989 there was already some confusion and so [RFC1122] clarified many things and extended the model. In 2004, [RFC3819] gave advice to link-layer protocol designers on a number of issues that affect upper layers and is the closest in intent to this document. Today's IP service model is not well documented in a single place, but is either implicit or discussed piecemeal in many different RFCs. As a result, today's IP service model is actually not well known, or at least is often misunderstood.

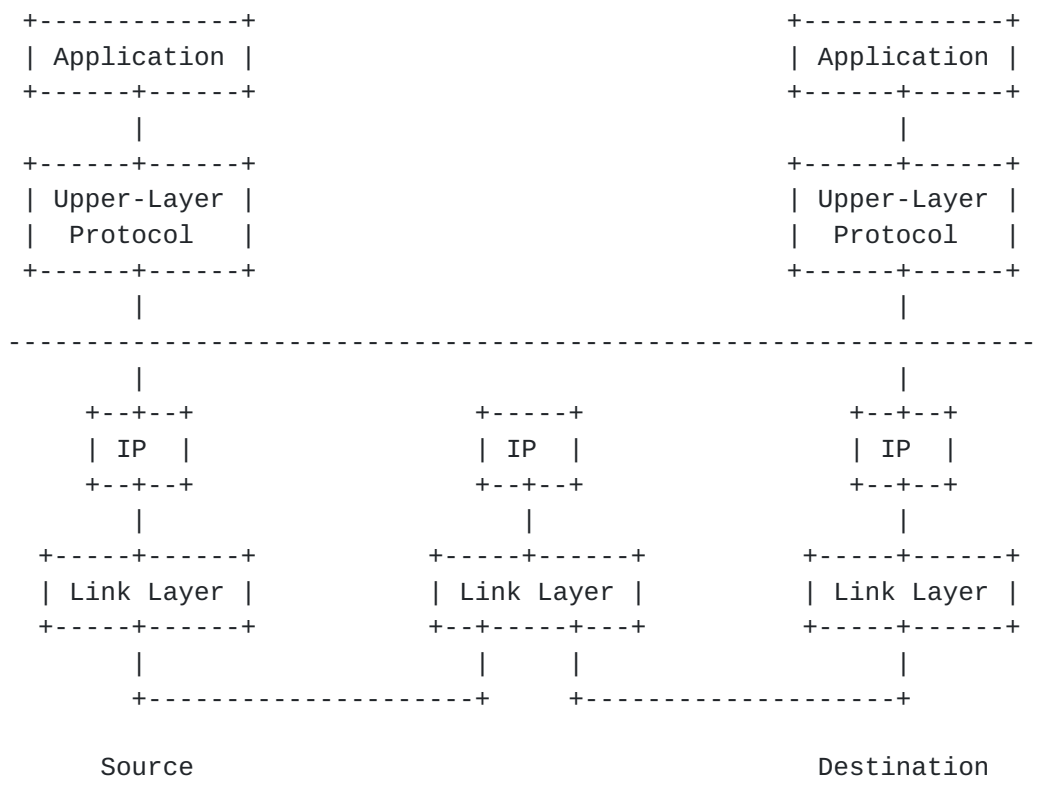
In the early days of IP, changing or extending the basic IP service model was easier since it was not as widely deployed and there were fewer implementations. Today, the ossification of the Internet makes evolving the IP model even more difficult. Thus it is important to understand the evolution of the IP model for two reasons:

1. To make it clear what upper-layer protocols and applications can and cannot depend on. There are many misconceptions on which applications may be based, and which are problematic.
2. To document lessons for future evolution to take into account. It is important that the service model remain consistent, rather than evolving in two opposing directions. It is sometimes the case in IETF Working Groups today that directions are considered or even taken which would change the IP service model. Doing this without understanding the implications on applications can be dangerous.

This draft attempts to document various aspects of the IP service model and how it has evolved over time. In particular, it attempts to document the properties of the IP layer, as seen by upper-layer protocols and applications, especially properties that were (and at times still are) incorrectly perceived to exist. It also highlights properties which would cause problems if changed.

## 2. The IP Service Model

In this document, we use the term "IP Service Model" to refer to the model exposed by IP to higher-layer protocols and applications. This is depicted in Figure 1 by the horizontal line.




IP Service Model

Figure 1

The foundation of the IP service model today is documented in [RFC0791] section 2.2. Generally speaking, IP provides a connectionless delivery service for variable size packets, which does not guarantee ordering, delivery, or lack of duplication, but is merely best effort (although some packets may get better service than others). Senders can send to a destination address without signaling a priori, and receivers just listen on an already provisioned address, without signaling a priori.

Architectural principles of the IP model are further discussed in [RFC1958] and in [NEWARCH] sections 5 and 6.

## 2.1. Links and Subnets

Section 2.1 of [RFC4903] discusses the terms "link" and "subnet" with respect to the IP model.

A "link" in the IP service model refers to the topological area within which a packet with an IPv4 TTL or IPv6 Hop Limit of 1 can be delivered. That is, where no IP-layer forwarding (which entails a TTL/Hop Limit decrement) occurs between two nodes.

A "subnet" in the IP service model refers to the topological area within which addresses from the same subnet prefix are assigned to interfaces.

## 3. Common Application Misconceptions

Below is a list of properties which are often assumed by applications and upper-layer protocol, but which have become less true over time.

### 3.1. Misconceptions about routing

#### 3.1.1. Claim: Reachability is symmetric

Many applications assume that if a host A can contact a host B, then the reverse is also true. Examples of this behavior include request-response patterns, which require reverse reachability only after forward reachability, as well as callbacks (e.g., as used by the File Transfer Protocol (FTP) [RFC0959]).

Originally it was the case that reachability was symmetric (although the path taken may not be), both within a link and across the Internet. With the advent of technologies such as Network Address Translators (NATs) and firewalls (as in the following example figure), this can no longer be assumed. Today, host-to-host connectivity is challenging if not impossible in general. It is relatively easy to initiate communication from hosts (A-E in the example diagram) to servers (S), but not vice versa, nor between hosts A-E. For a longer discussion on peer-to-peer connectivity see [RFC5694] Appendix A.

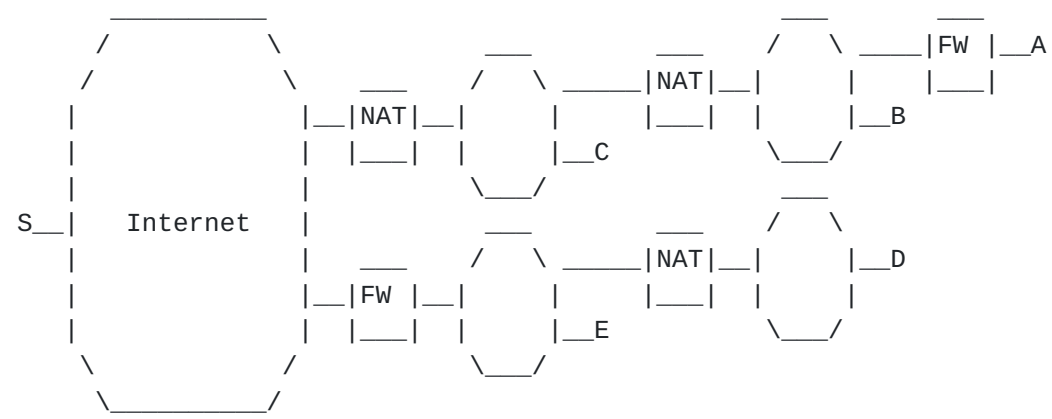



Figure 2

However, it is still the case that if a request can be sent, then a reply to that request can generally be received, but an unsolicited

request in the other direction may not be received. [RFC2993] discusses this in more detail.

There are also links (e.g., satellite) that were defined as unidirectional links and hence an address on such a link results in asymmetric reachability. [RFC3077] explicitly addresses this problem for multi-homed hosts by tunneling packets over another interface in order to restore symmetric reachability.

Finally, even with common wireless networks such as 802.11, this assumption may not be true, as discussed in [WIRELESS] section 5.5.

### 3.1.2. Claim: Reachability is transitive

Many applications assume that if a host A can contact host B, and B can contact C, then host A can contact C. Examples of this behavior include applications and protocols that use referrals.

Originally it was the case that reachability was transitive, both within a link and across the Internet. With the advent of technologies such as NATs and firewalls and various routing policies, this can no longer be assumed across the Internet, but it is often still true within a link. As a result, upper-layer protocols and applications may be relying on transitivity within a link. However, some radio technologies, such as 802.11 ad-hoc mode, violate this assumption within a link.

### 3.1.3. Claim: Error messages can be received in response to data packets

Some upper-layer protocols and applications assume that ICMP error messages will be received in response to packets sent that cannot be delivered. Examples of this include the use of Path MTU Discovery [RFC1191][RFC1981]) relying on messages indicating packets were too big, and traceroute and the use of expanding ring search [RFC1812] relying on messages indicating packets reached their maximum hop count limit.

Originally this assumption largely held, but many ICMP senders then chose to rate-limit responses in order to mitigate denial-of-service attacks, and many firewalls now block ICMP messages entirely. For a longer discussion, see [RFC2923] section 2.1.

This led to an alternate mechanism for Path MTU Discovery that does not rely on this assumption being true [RFC4821], and guidance to firewall administrators ([RFC2979] section 3.1.1 and [RFC4890]).

### 3.1.4. Claim: Multicast is supported within a link

[RFC1112] introduced multicast to the IP service model. In this evolution, senders still just send to a destination address without signaling a priori, but in contrast to the original IP model, receivers must signal to the network before they can receive traffic to a multicast address.

Today, many applications and protocols use multicast addresses, including protocols for address configuration, service discovery, etc. (See [MCAST4] and [MCAST6] for those that use well-known addresses.)

Most of these only assume that multicast works within a link, and may or may not function across a wider area. While network-layer multicast works over most link types, there are Non-Broadcast Multi-Access (NBMA) links over which multicast does not work (e.g., X.25, ATM, frame relay, 6to4, ISATAP, Teredo) and this can interfere with some protocols and applications. Similarly, there are links such as 802.11 ad-hoc mode where multicast packets may not get delivered to all receivers on the link. [RFC2461] and its successor [RFC4861] both state:

> "Note that all link types (including NBMA) are expected to provide multicast service for applications that need it (e.g., using multicast servers)."

However, not all link types today meet this expectation.

### 3.1.5. Claim: IPv4 broadcast is supported

IPv4 broadcast support was originally defined on a link, across a network, and for subnet directed broadcast, and is used by many applications and protocols. For security reasons, however, [RFC2644] deprecated forwarding of broadcast packets. Thus, since 1999, broadcast can only be relied on within a link. Still, there exist NBMA links over which broadcast does not work, and there exist some "semi-broadcast" links (e.g., 802.11 ad-hoc mode) where broadcast packets may not get delivered to all nodes on the link. Another case where broadcast fails to work is when a /32 or /31 is assigned to a point-to-point interface (e.g., [RFC3021]), leaving no broadcast address available.

The addition of link-scoped multicast to the IP service model to a large extent obsoleted the need for broadcast. It is also worth noting that the broadcast API model used by most platforms allows receivers to just listen on an already provisioned address, without signaling a priori, but in contrast to the unicast API model, senders must signal to the local IP stack (SO_BROADCAST) before they can send

traffic to a broadcast address. However, from the network's perspective, the host still sends without signaling a priori.

### 3.1.6. Claim: Multicast/broadcast is less expensive than replicated unicast

Some applications and upper-layer protocols that use multicast or broadcast do so not because they do not know the addresses of receivers, but simply to avoid sending multiple copies of the same packet over the same link.

In wired networks, sending a single multicast packet on a link is generally less expensive than sending multiple unicast packets. This may not be true for wireless networks, where implementations can only send multicast at the basic rate, regardless of the negotiated rates of potential receivers. As a result, replicated unicast may achieve much higher throughput across such links than multicast/broadcast traffic.

### 3.1.7. Claim: The end-to-end latency of the first packet to a destination is typical

Many applications and protocols choose a destination address by sending a message to each of a number of candidates, picking the first one to respond, and then using that destination for subsequent communication. If the end-to-end latency of the first packet to each destination is atypical, this can result in a highly non-optimal destination being chosen, with much longer paths (and hence higher load on the Internet) and lower throughput.

Today, there are a number of reasons this is not true. First, when sending to a new destination there may be some startup latency resulting from the link-layer or network-layer mechanism in use, such as ARP resolution for instance. In addition, the first packet may follow a different path from subsequent packets. For example, protocols such as Mobile IPv6 [RFC3775], Protocol Independent Multicast - Sparse Mode (PIM-SM) [RFC4601], and the Multicast Source Discovery Protocol (MSDP) [RFC3618] send packets on one path, and then allow immediately switching to a shorter path, resulting in a large latency difference. There are various proposals currently being evaluated by the IETF Routing Research Group that result in similar path switching.

### 3.1.8. Claim: Reordering is rare

As discussed in [REORDER], [RFC2991], and [RFC3819] section 15, there are a number of effects of reordering. For example, reordering increases buffering requirements (and jitter) in many applications,

and in devices that do packet reassembly. TCP [RFC0793] in particular is adversely affected by reordering, since it enters fast-retransmit when three packets are received before a late packet, which drastically lowers throughput. Finally, some NATs and firewalls assume that the initial fragment arrives first, resulting in packet loss when this is not the case.

Today there are number of things that cause reordering. For example, some routers do per-packet round-robin load balancing, which, depending on the topology, can result in a great deal of reordering. As another example, when a packet is fragmented at the sender, some hosts send the last fragment first. Finally, as discussed in Section 3.1.7, protocols that do path switching after the first packet result in deterministic reordering within the first burst of packets.

### 3.1.9. Claim: Loss is rare and probabilistic, not deterministic

In the original IP model, senders just send, without signaling the network a priori. This works to a degree. In practice, the last hop (and in rare cases, other hops) of the path needs to resolve next hop information (e.g., the link-layer address of the destination) on demand which results in queuing traffic, and if the queue fills up, some traffic gets dropped. This means that bursty sources can be problematic (and indeed a single large packet that gets fragmented becomes such a burst). The problem is rarely observed in practice today, either because the resolution within the last hop happens very quickly, or because bursty applications are rarer. However, any protocol that significantly increases such delays or adds new resolutions would be a change to the classic IP model and may adversely impact upper-layer protocols and applications that result in bursts of packets.

In addition, mechanisms that simply drop the first packet, rather than queuing it, also break this assumption. Similar to the result of reordering, they can result in a highly non-optimal destination being chosen by applications that use the first one to respond. Two examples of mechanisms that appear to do this are network interface cards that support a "Wake-on-LAN" capability where any packet that matches a specified pattern will wake up a machine in a power-conserving mode, but only after dropping the matching packet, and MSDP, where encapsulating data packets is optional, but doing so enables bursty sources to be accommodated while a multicast tree is built back to the source's domain.

### 3.1.10. Claim: An end-to-end path exists at a single point in time

In classic IP, applications assume that either an end-to-end path exists to a destination, or that the packet will be dropped. In addition, IP today tends to assume that the packet delay is relatively short (since the "Time"-to-live is just a hop count). In IP's earlier history, the TTL field was expected to also be decremented each second (not just each hop).

This assumption is still true in general today. However, the IRTF Delay Tolerant Networking Research Group is investigating ways for applications to use IP in networks where this assumption is not true, such as store-and-forward networks (e.g., packets carried by vehicles or animals).

### 3.1.11. Discussion

The reasons why assumptions listed above are increasingly less true can be divided into two categories: effects caused by attributes of link-layer technologies, and effects caused by network-layer technologies.

RFC 3819 [RFC3819] gives advice to link-layer protocol designers to minimize these effects. Generally the link-layer causes are not intentionally trying to break IP, but rather adding IP over the technology introduces the problem. Hence where the link-layer protocol itself does not do so, when specifying how IP is defined over such a link protocol, designers should compensate to the maximum extent possible. As examples, [RFC3077] and [RFC2491] compensate for lack of symmetric reachability and lack of link-layer multicast, respectively. That is, when IP is defined over a link type, the protocol designers should attempt to restore the assumptions listed in this document. For example, since an implementation can distinguish between 802.11 ad hoc mode vs. infrastructure mode, it may be possible to define a mechanism below IP to compensate for the lack of transitivity over such links.

At the network layer, as a general principle, we believe that reachability is good. For security reasons ([RFC4948]), however, it is desirable to restrict reachability by unauthorized parties; indeed IPsec, an integral part of the IP model, provides one means to do so. Where there are issues with asymmetry, non-transitivity, and so forth, which are not direct results of restricting reachability to only authorized parties (for some definition of authorized), the IETF should attempt to avoid or solve such issues. Similar to the principle outlined in [RFC1958] section 3.9, the general theme when defining a protocol is to be liberal in what effects you accept, and conservative in what effects you cause.

However, in being liberal in what effects you accept, it is also important to remember that diagnostics are important, and being too liberal can mask problems. Thus a tussle exists between the desire to provide a better experience to one's own users or applications and thus be more successful ([RFC5218]), vs. the desire to put pressure on getting problems fixed. One solution is to provide a separate "pedantic mode" that can be enabled to see the problems rather than mask them.

## 3.2. Misconceptions about addressing

### 3.2.1. Claim: Addresses are stable over long periods of time

Originally addresses were manually configured on fixed machines, and hence addresses were very stable. With the advent of technologies such as DHCP, roaming, and wireless, addresses can no longer be assumed to be stable for long periods of time ([RFC3775] section 4.2). However, the APIs provided to applications today typically still assume stable addresses (e.g., address lifetimes are not exposed to applications that get addresses). This can cause problems today when addresses become stale.

For example, many applications resolve names to addresses and then cache them without any notion of lifetime. In fact, the classic name resolution APIs do not even provide applications with the lifetime of entries.

Proxy Mobile IPv6 [RFC5213] tries to restore this assumption to some extent by preserving the same address while roaming around a local area. The issue of roaming between different networks has been known since at least 1980 when [IEN135] proposed a mobility solution that attempted to restore this assumption by adding an additional address that can be used by applications which is stable while roaming anywhere with Internet connectivity. More recent protocols such as Mobile IPv6 (MIP6) [RFC3775] and the Host Identity Protocol (HIP) [RFC4423] follow in this same vein.

### 3.2.2. Claim: An address is four bytes long

Many applications and protocols were designed to only support addresses that are four bytes long. Although this was sufficient for IPv4, the advent of IPv6 made this assumption invalid and with the exhaustion of IPv4 address space this assumption will become increasingly less true. There have been some attempts to try to mitigate this problem with limited degrees of success in constrained cases. For example, "Bump-In-the-Stack" [RFC2767] and "Bump-in-the-API" [RFC3338] attempt to provide four-byte "IPv4" addresses for IPv6 destinations, but have many limitations including (among a number of

others) all the problems of NATs.

### 3.2.3. Claim: A host has only one address on one interface

Although many applications assume this (e.g., by calling a name resolution function such as gethostbyname and then just using the first address returned), it was never really true to begin with, even if it was the common case. Even [RFC0791] states:

> "provision must be made for a host to have several physical interfaces to the network with each having several logical internet addresses".

However today this assumption is increasingly less true, with the advent of multiple interfaces (e.g., wired and wireless), dual-IPv4/IPv6 nodes, multiple IPv6 addresses on the same interface (e.g., link-local and global), etc. Similarly, many protocol specifications such as DHCP only describe operations for a single interface, whereas obtaining host-wide configuration from multiple interfaces presents a merging problem for nodes in practice. Too often this problem is simply ignored by Working Groups, and applications and users suffer as a result from poor merging algorithms.

One use of protocols such as MIP6 and HIP is to make this assumption somewhat more true by adding an additional "address" that can be the one used by such applications, and the protocol will deal with the complexity of multiple physical interfaces and addresses.

### 3.2.4. Claim: A non-multicast/broadcast address identifies a single host over a long period of time

Many applications and upper-layer protocols maintain a communication session with a destination over some period of time. If that address is reassigned to another host, or if that address is assigned to multiple hosts and the host at which packets arrive changes, such applications can have problems.

In addition, many security mechanisms and configurations assume that one can block traffic by IP address, implying that a single attacker can be identified by IP address. If that IP address can also identify many legitimate hosts, apply such a block can result in denial-of-service.

[RFC1546] introduced the notion of anycast to the IP service model. It states:

> Because anycasting is stateless and does not guarantee delivery of multiple anycast datagrams to the same system, an application cannot be sure that it is communicating with the same peer in two successive UDP transmissions or in two successive TCP connections

to the same anycast address.
The obvious solutions to these issues are to require applications which wish to maintain state to learn the unicast address of their peer on the first exchange of UDP datagrams or during the first TCP connection and use the unicast address in future conversations.

The issues with anycast are further discussed in [RFC4786] and [I-D.iab-anycast-arch-implications].

Another mechanism by which multiple hosts use the same address is as a result of scoped addresses, as defined for both IPv4 [RFC1918] [RFC3927] and IPv6 [RFC4007]. Because such addresses can be reused within multiple networks, hosts in different networks can use the same address. As a result, a host that is multihomed to two such networks cannot use the destination address to uniquely identify a peer. For example, a host can no longer use a 5-tuple to uniquely identify a TCP connection. This is why IPv6 added the concept of a "zone index".

Yet another example is that, in some high-availability solutions, one host takes over the IP address of another failed host.

See [RFC2101], [RFC2775], and [I-D.ietf-intarea-shared-addressing-issues] for additional discussion on address uniqueness.

### 3.2.5. Claim: An address can be used as an indication of physical location

Some applications attempt to use an address to infer some information about the physical location of the host with that address. For example, geo-location services are often used to provide targeted content or ads.

Various forms of tunneling have made this assumption less true, and this will become increasingly less true as the use of IPv4 NATs for large networks continues to increase. See [I-D.ietf-intarea-shared-addressing-issues] section 7 for a longer discussion.

### 3.2.6. Claim: An address used by an application is the same as the address used for routing

Some applications assume that the address the application uses is the same as that used by routing. For example, some applications use raw sockets to read/write packet headers, including the source and destination addresses in the IP header. As another example, some

applications make assumptions about locality (e.g., whether the destination is on the same subnet) by comparing addresses.

Protocols such as Mobile IPv6 and HIP specifically break this assumption (in an attempt to restore other assumptions as discussed above). Recently, the IRTF Routing Research Group has been evaluating a number of possible mechanisms, some of which would also break this assumption, while others preserve this assumption near the edges of the network and only break it in the core of the Internet.

Breaking this assumption is sometimes referred to as an "identifier/ locator" split. As originally defined in 1978 ([IEN019], [IEN023]), however, an address was originally defined as only a locator, whereas names were defined to be the identifiers. However, the TCP protocol then used addresses as identifiers.

Finally, in a liberal sense, any tunneling mechanism might be said to break this assumption, although in practice applications that make this assumption will continue to work. Since the address of the inside of the tunnel is still used for routing as expected.

### 3.2.7. Claim: A subnet is smaller than a link

In the classic IP model, a "subnet" is smaller than, or equal to, a "link". Destinations with addresses in the same on-link subnet prefix can be reached with TTL (or Hop Count) = 1. Link-scoped multicast packets, and all-ones broadcast packets will be delivered (in a best effort fashion) to all listening nodes on the link. Subnet broadcast packets will be delivered (in a best effort fashion) to all listening nodes in the subnet. There have been some efforts in the past (e.g., [RFC0925], [RFC3069]) to allow multi-link subnets and change the above service model, but the adverse impact on applications that have such assumptions recommend against changing this assumption.

[RFC4903] discusses this topic in more detail and surveys a number of protocols and applications that depend on this assumption. Specifically, some applications assume that, if a destination address is in the same on-link subnet prefix as the local machine, then therefore packets can be sent with TTL=1, or that packets can be received with TTL=255, or link-scoped multicast or broadcast can be used to reach the destination.

### 3.2.8. Claim: Selecting a local address selects the interface

Some applications assume that binding to a given local address constrains traffic reception to the interface with that address, and that traffic from that address will go out on that address's

interface. However, [RFC1122] section 3.3.4.2 defines two models: the Strong End System (or Strong host) model where this is true, and the Weak End System (or Weak host) model where this is not true. In fact any router is inherently a weak host implementation, since packets can be forwarded between interfaces.

### 3.2.9. Claim: An address is part of an on-link subnet prefix

To some extent, this was never true, in that there were cases in IPv4 where the "mask" was 255.255.255.255, such as on a point-to-point link where the two endpoints had addresses out of unrelated address spaces, and no on-link subnet prefix exists on the link. However, this didn't stop many platforms and applications from assuming that every address had a "mask" (or prefix) that was on-link. The assumption of whether a subnet is on-link (in which case one can send directly to the destination after using ARP/ND) or off-link (in which case one just sends to a router) has evolved over the years, and it can no longer be assumed that an address has an on-link prefix. In 1998, [RFC2461] introduced the distinction as part of the core IPv6 protocol suite. This topic is discussed further in [I-D.wbeebee-on-link-and-off-link-determination], and [RFC4903] also touches on this topic with respect to the service model seen by applications.

### 3.2.10. Discussion

RFC 1958 [RFC1958] section 4.1 states: "In general, user applications should use names rather than addresses."

We emphasize the above point, which is too often ignored. Many commonly used APIs unnecessarily expose addresses to applications that already use names. Similarly, some protocols are defined to carry addresses, rather than carrying names (instead of or in addition to addresses). Protocols and applications that are already dependent on a naming system should be designed in such a way that they avoid or minimize any dependence on the notion of addresses.

One challenge is that many hosts today do not have names that can be resolved. For example, a host may not have a fully-qualfied domain name (FQDN) or a Domain Name System (DNS) server that will host its name.

Applications that, for whatever reason, cannot use names should be IP-version agnostic.

## 3.3. Misconceptions about upper-layer extensibility

### 3.3.1. Claim: New transport-layer protocols can work across the Internet

IP was originally designed to support the addition of new transport-layer protocols, and [PROTOCOLS] lists many such protocols.

However, as discussed in [I-D.rosenberg-internet-waist-hourglass], NATs and firewalls today break this assumption and often only allow UDP and TCP (or even just HTTP).

Hence while new protocols may work from some places, they will not necessarily work from everywhere, such as from behind such NATs and firewalls.

Since even UDP and TCP may not work from everywhere, it may be necessary for applications to support "HTTP failover" modes. The use of HTTP as a "transport of last resort" has become common (e.g., [BOSH] among others) even in situations where it is sub-optimal, such as in real-time communications or where bi-directional communications is required. Also, the IETF HyBi Working Group is now in the process of designing a standards-based solution for layering other protocols on top of HTTP. As a result of having to support HTTP failover, applications may have to be engineered to sustain higher latency.

### 3.3.2. Claim: If one stream between a pair of addresses can get through, then so can another

Some applications and protocols use multiple upper-layer streams of data between the same pair of addresses, and initiated by the same party. Passive-mode FTP [RFC0959], and RTP [RFC3550], are two examples of such protocols, which use separate streams for data vs. control channels.

Today, there are many reasons this may not be true. Firewalls, for example, may selectively allow/block specific protocol numbers and/or values in upper-layer protocol fields (such as port numbers). Similarly, middleboxes such as NATs that create per-stream state may cause other streams to fail once they run out of space to store additional stream state.

### 3.3.3. Discussion

[NEWARCH] section 5.1 discusses the primary requirements of the original internet architecture, including Service Generality. It states:

"This goal was to support the widest possible range of applications, by supporting a variety of types of service at the transport level. Services might be distinguished by speed, latency, or reliability, for example. Service types might include virtual circuit service, which provides reliable, full-duplex byte streams, and also datagram service, which delivers individual packets with no guarantees of reliability or ordering. The requirement for datagram service was motivated by early ARPAnet experiments with packet speech (using IMP Type 3 messages)."

The reasons the assumptions in this section are becoming less true are due to network-layer (or higher-layer) techniques being introduced that interfere with the original requirement. Generally these are done either in the name of security, or as a side effect of solving some other problem such as address shortage. Work is needed to investigate ways to restore the original behavior while still meeting today's security requriements.

## 3.4. Misconceptions about security

### 3.4.1. Claim: Packets are unmodified in transit

Some applications and upper-layer protocols assume that a packet is unmodified in transit, except for a few well-defined fields (e.g., TTL). Examples of this behavior include protocols that define their own integrity protection mechanism such as a checksum.

This assumption is broken by NATs as discussed in [RFC2993] and other middleboxes that modify the contents of packets. There are many tunneling technologies (e.g., [RFC4380]) that attempt to restore this assumption to some extent.

The IPsec architecture [RFC4301] added security to the IP model, providing a way to address this problem without changing applications, although transport-mode IPsec is not currently widely used over the Internet.

### 3.4.2. Claim: Packets are private

The assumption that data is private has never really been true. However, many old applications and protocols (e.g., FTP) transmit passwords or other sensitive data in the clear.

IPsec provides a way to address this problem without changing applications, although it is not yet widely deployed, and doing encryption/decryption for all packets can be computationally expensive.

### 3.4.3. Claim: Source addresses are not forged

Most applications and protocols use the source address of some incoming packet when generating a response, and hence assume that it has not been forged (and as a result can often be vulnerable to various types of attacks such as reflection attacks).

Various mechanisms that restore this assumption include, for example, IPsec and Cryptographically Generated Addresses (CGAs) [RFC3972].

### 3.4.4. Discussion

A good discussion of threat models and common tools can be found in [RFC3552]. Protocol designers and applications developers are encouraged to be familiar with that document.

## 4. Security Considerations

This document discusses assumptions about the IP service model made by many applications and upper-layer protocols. Whenever these assumptions are broken, if the application or upper-layer protocol has some security-related behavior that is based on the assumption, then security can be affected.

For example, if an application assumes that binding to the IP address of a "trusted" interface means that it will never receive traffic from an "untrusted" interface, and that assumption is broken (as discussed in Section 3.2.8) then an attacker could get access to private information.

As a result, great care should be taken when expanding the extent to which an assumption is false. On the other hand, application and upper-layer protocol developers should carefully consider the impact of basing their security on any of the assumptions enumerated in this document.

It is also worth noting that many of the changes that have occurred over time (e.g., firewalls, dropping directed broadcasts, etc.) that are discussed in this document were done in the interest of improving security at the expense of breaking some applications.

## 5. IANA Considerations

This document has no IANA Actions.

## 6. Conclusion

Because a huge number of applications already exist that use TCP/IP for business-critical operations, any changes to the service model need to be done with extreme care. Extensions that merely add additional optional functionality without impacting any existing applications are much safer than extensions which change one or more of the core assumptions discussed above. Any changes to the above assumptions should only be done in accordance with some mechanism to minimize or mitigate the risks of breaking mission-critical applications. Historically, changes have been done without regard to such considerations and as a result the situation for applications today is already problematic. Key to maintaining an interoperable Internet is documenting and maintaining invariants that higher layers can depend on, and being very judicious with changes.

In general, lower-layer protocols should document the contract they provide to higher layers; that is, what assumptions the upper layer can rely on (sometimes this is done in the form of an applicability statement). Conversely, higher-layer protocols should document the assumptions they rely on from the lower layer (sometimes this is done in the form of requirements).

We must also recognize that a successful architecture often evolves as success brings growth and as technology moves forward. As a result, the various assumptions made should be periodically reviewed when updating protocols.

## 7. Acknowledgements

Chris Hopps, Dow Street, Phil Hallam-Baker, and others provided helpful discussion on various points that led to this document. Iain Calder, Brian Carpenter, Jonathan Rosenberg, Erik Nordmark, Alain Durand, and Iljitsch van Beijnum also provided valuable feedback.

## 8. IAB Members at the time of this writing

Loa Andersson
Gonzalo Camarillo
Stuart Cheshire
Russ Housley
Olaf Kolkman
Gregory Lebovitz
Barry Leiba
Kurtis Lindqvist
Andrew Malis
Danny McPherson
David Oran
Dave Thaler
Lixia Zhang

## 9. IAB Members at time of approval

Bernard Aboba
Marcelo Bagnulo
Ross Callon
Spencer Dawkins
Russ Housley
John Klensin
Olaf Kolkman
Danny McPherson
Jon Peterson
Andrei Robachevsky
Dave Thaler
Hannes Tschofenig

## 10. References

### 10.1. Normative References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC1112] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, August 1989.

[RFC1122] Braden, R., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, October 1989.

[RFC1546] Partridge, C., Mendez, T., and W. Milliken, "Host Anycasting Service", RFC 1546, November 1993.

[RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

[RFC2644] Senie, D., "Changing the Default for Directed Broadcasts in Routers", BCP 34, RFC 2644, August 1999.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4861] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)", RFC 4861, September 2007.

## 10.2. Informative References

[BOSH] Paterson, I., Smith, D., Saint-Andre, P., and J. Moffitt, "Bidirectional-streams Over Synchronous HTTP (BOSH)", XEP 0124, 2010, <http://xmpp.org/extensions/xep-0124.html>.

[I-D.iab-anycast-arch-implications] McPherson, D. and D. Oran, "Architectural Considerations of IP Anycast", draft-iab-anycast-arch-implications-00 (work in progress), February 2010.

[I-D.ietf-intarea-shared-addressing-issues] Ford, M., Boucadair, M., Durand, A., Levis, P., and P. Roberts, "Issues with IP Address Sharing", draft-ietf-intarea-shared-addressing-issues-02 (work in progress), October 2010.

[I-D.rosenberg-internet-waist-hourglass] Rosenberg, J., "UDP and TCP as the New Waist of the Internet Hourglass", draft-rosenberg-internet-waist-hourglass-00 (work in progress), February 2008.

[I-D.wbeebee-on-link-and-off-link-determination] Singh, H., Beebee, W., and E. Nordmark, "IPv6 Subnet Model", draft-wbeebee-on-link-and-off-link-determination-02 (work in progress), February 2008.

[IEN019] Shoch, J., "A note on Inter-Network Naming, Addressing, and Routing", IEN 19, January 1978, <ftp://ftp.rfc-editor.org/in-notes/ien/ien19.txt>.

[IEN023] Cohen, D., "On Names, Addresses and Routings", IEN 23, January 1978, <ftp://ftp.rfc-editor.org/in-notes/ien/ien23.txt>.

[IEN028] Postel, J., "Draft Internetwork Protocol Specification", IEN 28, February 1978, <ftp://ftp.rfc-editor.org/in-notes/ien/ien-index.html>.

[IEN135] Sunshine, C. and J. Postel, "Addressing Mobile Hosts in the ARPA Internet Environment", IEN 135, March 1980, <ftp://ftp.rfc-editor.org/in-notes/ien/ien135.txt>.

[MCAST4] Internet Assigned Numbers Authority, "IPv4 Multicast Addresses", <http://www.iana.org/assignments/multicast-addresses>.

[MCAST6] Internet Assigned Numbers Authority, "INTERNET PROTOCOL VERSION 6 MULTICAST ADDRESSES", <http://www.iana.org/assignments/ipv6-multicast-addresses>.

[NEWARCH] Clark, D., et al., "New Arch: Future Generation Internet Architecture", Air Force Research Laboratory Technical Report AFRL-IF-RS-TR-2004-235, August 2004, <http://www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA426770&Location=U2&doc=GetTRDoc.pdf>.

[PROTOCOLS] Internet Assigned Numbers Authority, "Protocol Numbers", <http://www.iana.org/assignments/protocol-numbers>.

[REORDER] Bennett, J., Partridge, C., and N. Shectman, "Packet reordering is not pathological network behavior", IEEE/ACM Transactions on Networking, Vol. 7, No. 6, December 1999.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC0925] Postel, J., "Multi-LAN address resolution", RFC 925, October 1984.

[RFC0959] Postel, J. and J. Reynolds, "File Transfer Protocol", STD 9, RFC 959, October 1985.

[RFC1191] Mogul, J. and S. Deering, "Path MTU discovery", RFC 1191, November 1990.

[RFC1812] Baker, F., "Requirements for IP Version 4 Routers",

RFC 1812, June 1995.

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

[RFC1958] Carpenter, B., "Architectural Principles of the Internet", RFC 1958, June 1996.

[RFC1981] McCann, J., Deering, S., and J. Mogul, "Path MTU Discovery for IP version 6", RFC 1981, August 1996.

[RFC2101] Carpenter, B., Crowcroft, J., and Y. Rekhter, "IPv4 Address Behaviour Today", RFC 2101, February 1997.

[RFC2491] Armitage, G., Schulter, P., Jork, M., and G. Harter, "IPv6 over Non-Broadcast Multiple Access (NBMA) networks", RFC 2491, January 1999.

[RFC2767] Tsuchiya, K., HIGUCHI, H., and Y. Atarashi, "Dual Stack Hosts using the "Bump-In-the-Stack" Technique (BIS)", RFC 2767, February 2000.

[RFC2775] Carpenter, B., "Internet Transparency", RFC 2775, February 2000.

[RFC2923] Lahey, K., "TCP Problems with Path MTU Discovery", RFC 2923, September 2000.

[RFC2979] Freed, N., "Behavior of and Requirements for Internet Firewalls", RFC 2979, October 2000.

[RFC2991] Thaler, D. and C. Hopps, "Multipath Issues in Unicast and Multicast Next-Hop Selection", RFC 2991, November 2000.

[RFC2993] Hain, T., "Architectural Implications of NAT", RFC 2993, November 2000.

[RFC3021] Retana, A., White, R., Fuller, V., and D. McPherson, "Using 31-Bit Prefixes on IPv4 Point-to-Point Links", RFC 3021, December 2000.

[RFC3069] McPherson, D. and B. Dykes, "VLAN Aggregation for Efficient IP Address Allocation", RFC 3069, February 2001.

[RFC3077] Duros, E., Dabbous, W., Izumiyama, H., Fujii, N., and Y. Zhang, "A Link-Layer Tunneling Mechanism for Unidirectional Links", RFC 3077, March 2001.

[RFC3338] Lee, S., Shin, M-K., Kim, Y-J., Nordmark, E., and A. Durand, "Dual Stack Hosts Using "Bump-in-the-API" (BIA)", RFC 3338, October 2002.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

[RFC3618] Fenner, B. and D. Meyer, "Multicast Source Discovery Protocol (MSDP)", RFC 3618, October 2003.

[RFC3775] Johnson, D., Perkins, C., and J. Arkko, "Mobility Support in IPv6", RFC 3775, June 2004.

[RFC3819] Karn, P., Bormann, C., Fairhurst, G., Grossman, D., Ludwig, R., Mahdavi, J., Montenegro, G., Touch, J., and L. Wood, "Advice for Internet Subnetwork Designers", BCP 89, RFC 3819, July 2004.

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

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, March 2005.

[RFC4007] Deering, S., Haberman, B., Jinmei, T., Nordmark, E., and B. Zill, "IPv6 Scoped Address Architecture", RFC 4007, March 2005.

[RFC4380] Huitema, C., "Teredo: Tunneling IPv6 over UDP through Network Address Translations (NATs)", RFC 4380, February 2006.

[RFC4423] Moskowitz, R. and P. Nikander, "Host Identity Protocol (HIP) Architecture", RFC 4423, May 2006.

[RFC4601] Fenner, B., Handley, M., Holbrook, H., and I. Kouvelas, "Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised)", RFC 4601, August 2006.

[RFC4786] Abley, J. and K. Lindqvist, "Operation of Anycast Services", BCP 126, RFC 4786, December 2006.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

[RFC4890] Davies, E. and J. Mohacsi, "Recommendations for Filtering ICMPv6 Messages in Firewalls", RFC 4890, May 2007.

[RFC4903] Thaler, D., "Multi-Link Subnet Issues", RFC 4903, June 2007.

[RFC4948] Andersson, L., Davies, E., and L. Zhang, "Report from the IAB workshop on Unwanted Traffic March 9-10, 2006", RFC 4948, August 2007.

[RFC5213] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, August 2008.

[RFC5218] Thaler, D. and B. Aboba, "What Makes For a Successful Protocol?", RFC 5218, July 2008.

[RFC5694] Camarillo, G. and IAB, "Peer-to-Peer (P2P) Architecture: Definition, Taxonomies, Examples, and Applicability", RFC 5694, November 2009.

[WIRELESS] Kotz, D., Newport, C., and C. Elliott, "The mistaken axioms of wireless-network research", Dartmouth College Computer Science Technical Report TR2003-467, July 2003, <http://pdos.csail.mit.edu/decouto/papers/kotz03.pdf>.

## Author's Address

Dave Thaler
IAB
One Microsoft Way
Redmond, WA 98052
USA

Phone: +1 425 703 8835
Email: dthaler@microsoft.com