A Group-Centric Internet Architecture

First Year Report

Stephen D. Strowes

Abstract

The currently deployed Internet architecture is facing serious problems which must be solved to ensure continued operation: the IPv4 address space is rapidly running out, and the forwarding tables in the default-free zone of the routing infrastructure are growing at unsustainable, super-linear rates. The problems are such that it is worthwhile to revisit the architecture of the Internet, in light of the fact that application usage is now predominantly group-based, and significantly different to expected usage during design and deployment of the current set of widely used Internet protocols.

This report supports this assertion, provides background information on the current Internet architecture and related work, and sketches some initial concepts for a group-centric Internet architecture.

Contents

List of Figures

Chapter 1

Introduction

The Internet consists of a collection of interlinked, autonomous, heterogeneous networks collaborating to move packeted data between endpoints so as to provide the best-effort packet-forwarding Internet service we are familiar with today. The general packet-switched nature of the network allows for wide variation in types of applications which can make use of it.

Deployment of the current Internet architecture began around 1982, and there have been few fundamental changes since. The architecture was designed to support resource sharing, such that networked computers could be controlled remotely via remote shells, and commonly used (and often expensive) hardware such as printers could be shared.

However, network usage has changed over time. As the size of the network has increased, so too has the population of the network, as has the availability of cheap hardware and fast network connections. In this chapter I will cover common network usage scenarios demonstrating this in §1.1, followed by a brief discussion on the lifetime of the current architecture §1.2. I will finalise this chapter with my thesis statement in §1.3 and an outline of the remainder of this report in §1.4.

1.1 Network Usage Models

The Internet has spawned many applications not yet imagined when the network protocols were designed; many of these applications require communication between groups of hosts on the network, and yet these take place primarily over point-to-point unicast connections.

Examples of such commonly used applications include:

1. File-sharing, which often occurs in large groups, and requires reliable, but not low-latency, delivery of packets. The actual process of sharing file data may occur over groups of highly variable sizes, from very small to very large.
2. Email mailing lists or newsgroups require reliable, but not time critical, delivery. This application may also expect groups of highly variable size, from zero subscribers to many thousands.

3. RSS feeds, or any other periodically refreshed data to be best pushed to interested end-hosts or intermediaries, such as weather reports, traffic updates, operating system updates, etc. Delivery must be reliable, but timely delivery is not necessarily a requirement. Again, this application exhibits highly variable group sizes, though group size may vary slowly. Suggests potentially infrequent bursts of high-volume data transfer.

4. Web access could benefit from multicasting frequently accessed pages or data out to recipients [3]. Such a mechanism would ease the bandwidth requirements of web hosts. Delivery must be reliable, but isn't time critical.

5. News/stock tickers would require reliable, delay invariant delivery of data to all end-points or intermediaries. Delay invariant transports for group communication have been researched [4, 5].

6. Audio/Video conferencing is a natural candidate for a group-based architecture; best-effort delivery generally suits these applications, and low-latency packet delivery is preferred between sources and destinations. Group size can vary from small (2 or 3 people) to very large (conferences, possibly with mechanisms in place to entitle group members to ask questions; many audience members may be passive observers.)

7. Online gaming often has high bandwidth requirements for providers of the gaming platform, and this application could certainly make use of a group-centric Internet. Such groups require low-latency packet delivery and, in many cases, best-effort delivery. Often, but not always, small groups and groups may evolve over longer periods of time.

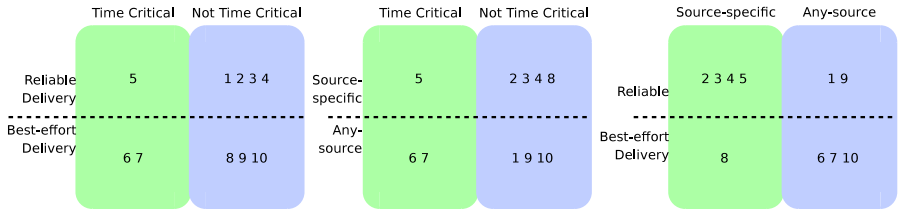
8. IPTV, and other media streaming services, involve one or a handful of sources distributing data often into a large receiver group. The receiver group may expand and contract rapidly, depending on the nature of the media being served. Like audio/video conferencing, best-effort packet delivery is good enough. Delivery probably isn't time-critical; latency isn't a major problem for most IPTV services, in the sense that a IPTV stream could arrive even a few seconds late without problems.

9. Instant messaging, IRC, etc, require reliable delivery of messages, but, again, isn't necessarily a time critical service. These generally exhibit small, often transient groups.

10. Resource discovery on local networks already uses multicast, for good reason: well-known group addresses can be "hardwired" through standardisation of discovery protocols (e.g., the Service Location Protocol, SLP), and resource discovery naturally lends itself to locating *nearby* resources. A best-effort packet delivery service is probably acceptable for most applications in this category.

We clearly have a variety of Internet applications which exhibit different group characteristics and delivery requirements, as noted in the various groupings in Fig. 1.1. These all, however, demonstrate some form of group-based communication, and certainly comprise the majority of network traffic today.

However, the current Internet has not seen widespread deployment of a multicast service which would benefit these applications. In order to provide some of these services,



(a) Timely delivery against reliability.

(b) Timely delivery against group model.

(c) Group model against reliability.

Figure 1.1: Characteristics of various application use cases.

dedicated service providers often have to deploy expensive centralised server farms to and act as a proxy for group members; other services require expensive volumes of bandwidth, or hire dedicated data distribution companies[1] to alleviate the cost of providing infrastructure. Other services are built using complex network overlays to provide multicast capabilities which they cannot be reliably used in the Internet.

It becomes clear that a typical Internet usage pattern often involves some form of group communication, in some cases for the purposes network efficiency in data distribution and in some cases for real-time group communication. Yet the current underlying network is designed on a point-to-point principle, with multicast still not quite available in the Internet.

1.2 Lifetime of Current Internet

In addition to the evolution of available applications as the network population has changed, the current network architecture is facing scalability issues in the core of the network, with forwarding tables expanding at super-linear rates, which may affect the forwarding capabilities of the routers in the default-free zone (DFZ) of the network.

The IPv4 address space is also running short, and is due to run out within a few years. IPv6 deployment has steadily increased, but is still largely unused; IPv6 addresses, being larger than IPv4 addresses, will also exacerbate the scalability issues in the forwarding tables of the DFZ. There are various causes for the poor scaling characteristics of the network, covered in §3.4.

1.3 Thesis Statement

Alternative network architectures have been proposed that focus on altering unicast communication either by decoupling endpoint identification from network location, or by relieving the scaling properties of BGP routing tables; generally, multicast communication is left as an afterthought.

I assert that a new Internet architecture may be designed in which group communication is the network's fundamental primitive, and that such an architecture would simplify

[1] For example, Akamai, `http://www.akamai.com`.

application design and allow for new applications to be brought to the network without expensive overlay infrastructures.

I will demonstrate this by designing an alternative architecture which encapsulates the aim of supporting group-centric communication, based on the existing principles of datagram-based networks (e.g., the end-to-end argument), with discussion on state placement within the network such that supporting groups is feasible. The feasibility of such an architecture will be demonstrated by building and testing a networked system using the new architecture, which will be used to test protocols and evaluate the control-plane overhead required to offer the group-centric service.

1.4 Report Outline

In the remainder of this report, I will cover various general principles which underpin network design in Chapter 2. In Chapters 3 and 4 I will cover the deployed Internet architecture and other alternative Internet architectures which have been published respectively. In Chapter 5, I will propose some requirements and initial ideas for a group-centric network architecture based on the various principles outlined and the previous work. I will finalise this report in Chapter 6 with a brief summary on the direction this work may go.

Chapter 2

Design Principles for Network Architectures

A network architecture determines how nodes are addressed, what state is required (and where) to enable communication between endpoints, the routing algorithms employed to generate forwarding tables, the semantics of the data transfer mechanism, and so forth. Any architecture makes certain assumptions of the underlying physical infrastructure (such as delivery reliability) and influences the behaviour of the transport protocols which run on top of that architecure (such as how a lack of reliability is handled at communicating nodes).

A network architecture aiming to provide a global service, i.e., to operate as an *inter*-network, must consider how it creates a network of networks, and what is required to achieve this. Implied in this requirement is that the architecture is capable of running in a heterogeneous networked environment, bridging between independently controlled networks of differing types. Each one of these independent networks is termed an *autonomous system*, an AS. This implies a two-tier approach to routing: intra-AS routing, and inter-AS routing.

Aside from ensuring interoperability, the network architecture also defines what semantics for data transmission networked nodes should expect, and where state must be stored in the network to provide those semantics.

This chapter discusses some of the high-level principles, and the differing viewpoints with regards to those principles, which drives actual design and implementation of network architectures for general computer communication. Arguably, a consistent set of network principles help further development of that architecture. In this chapter, I discuss methods of data transfer between nodes in §2.1. I consider the end-to-end argument for function placement in the network in §2.2, and use this argument in the context of network protocol stack layering in §2.3. I look at the differing principles of network identification and location in §2.4. In §2.5, I will discuss the importance of clean design in terms of the evolvability of the network. These various discussions are primarily focussed on point to point communication, and tend not to focus on group communication; I will finalise this chapter with a discussion of these various principles in the context of group communication in §2.6.

2.1 Circuits and Datagrams

There are various broad categories for defining the communications characteristics of networks; these categories describe whether signalling state is required, how nodes are addressed in the network, and what information must be carried with the data stream to allow the network to deliver the data to its destination. These categories help define what we require of the network. Here, I consider three broad types: circuit switched networks, virtual circuit switched networks, and datagram switched networks.

2.1.1 Circuit Switching

Telephone networks have traditionally used circuit switching to achieve communication between endpoints. Communication in a circuit switched network requires that a call is set up prior to data being transferred between endpoints.

The call setup phase reserves bandwidth for the connection which, unless all of the reserved bandwidth is in constant use, is wasteful. Bandwidth is reserved until a call teardown to terminate the connection and relinquish the bandwidth.

Circuit switched networks are therefore best suited for constant bit rate applications which is rarely the case; consider the applications discussed in §1.1, and that these applications either involve intermittent publication of data, or will often utilise compression codecs which may introduce data rate variability.

In this environment, circuit switching is not the most suitable candidate for offering a general data network service with the various applications we might reasonably expect to run over it.

2.1.2 Virtual Circuit Switching

Similar to circuit switched networks, virtual circuit switched networks mimic the call setup/teardown phases required prior to any data transfer taking place: on setup, a path is chosen from source to destination, and state is placed in routers to allow for subsequent forwarding of packets between the two endpoints until that state is explicitly removed by a teardown once communication is complete. Virtual circuit packet switched networks do not necessarily offer the same bandwidth guarantees that circuit switched networks offer, and data is sent across the network in packetised form.

Once a virtual circuit has been formed, data packets are forwarded between endpoints on the basis of a small, fixed-size virtual circuit identifier, or label. This identifier is key to virtual circuit switching, allowing for packets to be passed between the endpoints with a relatively short header containing this identifier; the identifier is used to achieve a fast lookup for forwarding of packets.

The utilisation of this identifier to achieve a fast lookup over a forwarding table was intended to simplify the forwarding process and potentially help minimise jitter between packets. Further, short headers also provide a reasonable ratio for control to data bandwidth utilisation.

Virtual circuit switching implies a connection-oriented service with short, fixed-size circuit identifiers. To achieve circuit identifiers which are shorter than endpoint ad-

dresses, virtual circuit switching makes the assumption that the number of flows in the network will be much smaller than the number of endpoints. One advantage to setting up flows in this manners is that packets will arrive at their destination in order.

Virtual circuit switched networks have attracted much attention and seen widespread deployment; common examples of these are ATM and MPLS.

2.1.3 Datagram Switching

In contrast to virtual circuit switched networks, datagram packet switched networks do not require the setup/teardown signalling phases to initiate or terminate a virtual connection, and instead consider each datagram to be independent of all others. Thus, each datagram carries in its header at least a destination address, which the network uses at each point to forward the datagram toward its intended endpoint.

A datagram switched network therefore does not require per-connection state to be maintained in intermediate routers. The caveat is that processing of individual datagrams at intermediate nodes is a more expensive operation, potentially requiring a more complex lookup than a lookup with a circuit identifier. A key concern with packet switching is this potential additional processing overhead at intermediate nodes.

2.1.4 Discussion

The differences between these forms of data transmission suggest that the different models may be applicable in different settings.

Circuit switched networks guarantee bandwidth per-connection, but this guaranteed bandwidth would be wasteful given the variety of applications which may be run over a general communications network. Virtual circuit switching, with its requirement of state in intermediate nodes between endpoints but minus the bandwidth guarantees, is more applicable to applications which can justify that state. Datagram switching, however, only requires enough state in the network to forward datagrams toward endpoints, though forwarding decisions have to be made for each datagram received.

Traditional circuit switched networks, such as the telephone system, have allowed for "dumb" devices to communicate over the more intelligent network. At the other end of the spectrum, in a datagram switched network, it is the network which is perceived as "dumb", and the endpoints are more intelligent.

Determining which paradigm to use in a network architecture can be difficult. For example, all applications mentioned in §1.1 exhibit bursty traffic patterns, but it might be advantageous to set up forwarding state for certain applications dealing with real-time information, if there was clear benefit to doing so. On the other hand, applications exhibiting "bursty" traffic patterns might be better suited to a datagram switched networking, thus not requiring additional (and often dormant) state to be embedded in the network. For example, resource sharing and interactive sessions between hosts on a computer network neither exhibit a constant bit-rate characteristic, nor do they generate packets at predictably regular intervals justifying connection state.

Virtual connections can be emulated at the endpoints in a datagram switched network, albeit at a higher layer in the network stack (see §2.3). The benefit is that the underlying datagram forwarding infrastructure can adapt to reflect new or missing physical

links, and alter forwarding tables as appropriate, allowing for continued improvement of a connection between endpoints during the lifetime of that connection; this is much trickier to achieve in virtual circuit switched networks.

2.2 The End-to-End Argument and Fate Sharing

One of the key principles related to this discussion on state-placement underlying today's Internet is articulated in the end-to-end argument [6]. The end-to-end argument espouses minimal volumes of state within the network, and generality in the functions provided by the network. Increased complexity is then constructed at the endpoints to provide the functionality the applications require. The end-to-end argument is a discussion on what state belongs within a packet-switched network, which state to place only at endpoints, and why. Therefore, it is not an argument for no state in the network, but instead is an argument for some minimal set of common state in the network, if the benefits justify the cost; the argument asks us to evaluate and justify state stored within the network. For example, 100% reliability is not considered a requirement of the Internet, because the cost and complexity required to provide it in the network is much greater than achieving the same level of reliability by having endpoints retransmit missing packets.

This is also an argument for the generality of the network, and that functions should only be placed in the network if they are applicable to enough network traffic:

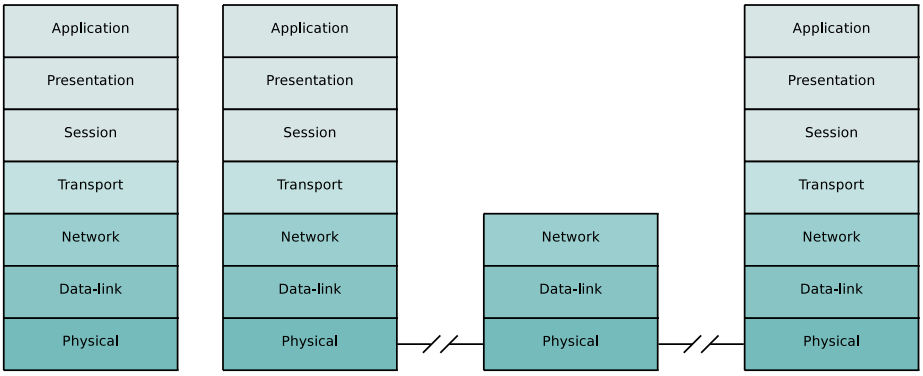
> "The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible." [6]

The end-to-end argument is very closely related to a similar argument for what is known as fate sharing [7]. The fate sharing argument adopts the stance of the application, rather that of the network, and suggests that the application should not have to rely on complex state replication in the network to maintain its communication with the another endpoint. In other words, if the communication terminates unexpectedly, then one of the endpoints is at fault, not the carrier between (barring a complete failure of the network).

The end-to-end and fate sharing arguments each suggest, in different ways, that the network functionality provided should be as simple a subset of common functionality as is feasible, with more complex functionality built at endpoints. This maps onto the argument for datagram switched networks in §2.1, with the "dumb" network and "intelligent" endpoints.

2.3 Protocol Layering and End-to-End Protocols

Explicit in the end-to-end argument is that state for an application be retained within that application, and that partial state is not needlessly placed within the network. The



(a) The full OSI model.

(b) The full OSI network stack at the endpoints, with a network-layer router in between.

Figure 2.1: The OSI layered protocol stack model.

argument implies some level of isolation between the different components in the network: the applications, and the network are two different entities, the application sitting atop the network. This isolation provides a protocol design pattern: layering.

Layering allows for compartmentalisation of functionality, such if that the network provides a best-effort datagram forwarding service, transport layer protocols can be deployed to enable reliable delivery of data, which applications can then use. These loosely-coupled layers should then be able to evolve reasonably independently of each other. The OSI Reference model [8], shown in Fig. 2.1(a), is an idealised model for network stack layering, but which does not specify details such as network layer addressing, etc.

The current network architecture (covered in detail in Chapter 3) is a connectionless, variable-length datagram switched network. Connections, concealing packets to the applications above, are created only via state at endpoints. TCP is an example of an end-to-end protocol operating at the transport layer (Fig. 2.1(b)), which is able to provide to the application something similar to virtual circuit channel as described in §2.1. The end-to-end principle of protocol design is best expressed thus:

> "An end-to-end protocol design should not rely on the maintenance of state (i.e. information about the state of the end-to-end communication) inside the network. Such state should be maintained only in the endpoints ... An immediate consequence of this is that datagrams are better than classical virtual circuits. The network's job is to transmit datagrams as efficiently and flexibly as possible. Everything else should be done at the fringes." [9]

However, strict layering of the protocol stack above the network layer is not a necessary design principle. While layering is a simple model for abstracting and isolating functionality between layers, some applications might benefit from directly handling all layers above the network layer, known as application level framing [10][1].

[1] Protocols such as RTP, and applications such NTE (the network text editor, [11]), wb (a shared whiteboard, [12]), make use of application level framing.

Other interesting work includes the Role-Based Architecture, which essentially generalises the stack such that a correct ordering of layers is maintained on a per-application basis, rather than via a general-purpose stack [13].

2.4 Network Locators and Endpoint Identifiers

In a datagram switched network with layered protocol stacks at endpoints, a distinction can be made between a *locator*, used by the network layer to route a packet from its source to its destination, possibly using some hierarchical addressing scheme or other such properties of the graph, and an *identifier*, used by the transport layer to uniquely identify an entity within the network no matter its locator.

The distinction between the two is obviously important in the context of mobility, but is also important in the modern market of service providers, whereby consumers (in this sense, any entity buying network connectivity) may wish to switch to a more competitive provider. Such a change of provider may affect the network locators required to find the consumer within the network, but should not affect the consumer's identity.

This distinction between an identifier, which can essentially be viewed as the property of the end-host, and a network locator, maps once again onto the end-to-end argument described in §2.2, and also maps neatly onto the principles of protocol layering identified in §2.3.

RFC 1498 [14] considers some of the terminology, and defines *names* (identifying the item of interest), *addresses* (describing an items location) and a *route* (identifying how to get to the item of interest). Other useful abstractions for a network architecture might be *users* or *groups*. There are also questions regarding the scope of identifiers, the uniqueness of identifiers, etc [15].

2.5 Evolvability

A key concern with any architecture is how to design it in such a way that it can be improved upon or replaced altogether; we cannot assume that network usage patterns, application behaviour, or business relationships between network operators will remain constant for the lifetime of the network, and so it seems unwise to not consider possible progression paths from one network architecture to another.

The current architecture has demonstrated difficulty in evolving from IPv4 to IPv6 despite a fairly wide deployment of IPv6. Part of this slow adoption is related to a lack of layer isolation, forcing higher layers to require modification to match the new network layer, and perhaps related to a perceived lack of gain in the new architecture. The co-existence of differing protocols is a complex, but required phase of an evolving network [16, 17].

Difficulty can be seen again in the deployment of IP Multicast, and solutions such as AMT [18], have been proposed which do not require total deployment of the service before it becomes useful, allowing for a progressive deployment of the new service by automatically tunnelling across the parts of the network which have not yet been enabled.

2.6 Discussion of Network Principles

In this chapter, I have covered some of the most basic of principles underpinning network architecture design. Most of these principles are not unicast-specific, it just so happens that the currently deployed Internet architecture describes a unicast-centric network. The architectural principles relating to the currently deployed Internet architecture are covered in [9, 19].

While most of these architectural principles apply quite clearly to point-to-point communication, fate sharing does not necessarily map into a group setting. The end-to-end argument discussed in §2.2 discusses function placement, and arguably functionality related to location of group members and forwarding packets to multiple receivers can be placed in the network by this argument, so can be applied to group communications.

Fate sharing in a group context, however, is more complex; it doesn't make sense for a node failure to inhibit communication between the remainder of a group. So, group-based transport mechanisms should be able to handle receipt of different subsets of packets from a single source, and should rely as little as possible on individual group members other than primary sources.

Multicast protocols for use in the Internet currently require additional on-path forwarding state on either a per-group or a per-source basis; this is somewhat closer to the ideas on virtual circuit switching presented in §2.1 than it is to datagram switching. This is not unfeasible, but a solution which retains a purer datagram switching principle may be cleaner, and more scalable.

If we make the argument that circuit switched networks do not suit the "bursty" nature of datagram-based traffic, then we're either suggesting that the unused bandwidth allocation is a negative, or that the state holding the connection is a negative. Thus, it becomes difficult to reasonably suggest an architecture which requires per-group (or per-source) forwarding state.

Chapter 3

The Internet Architecture

Prior to discussion on alternative Internet architectures, we must first consider the currently deployed architecture, how it handles multicast traffic, and its various shortcomings in a modern networked environment.

The expansion of the Internet since the inception of its various core protocols on which it relies has required those protocols to evolve over time. However, the generality of the protocols has, thus far, required that only reasonably small changes be made. In this chapter I will cover this historical perspective leading up to the current architecture in §3.1, before looking at the set of protocols which form the basis for the current architecture in §3.2 (for unicast protocols) and §3.3 (for multicast protocols).

Despite this evolution to the current architecture, there are various flaws now apparent in the deployed Internet architecture, with some trends suggesting that the network growth rate is not sustainable. Deployment of IP Multicast as a key service for group communication has also been slow. I will finalise this chapter with discussion of these problems in §3.4.

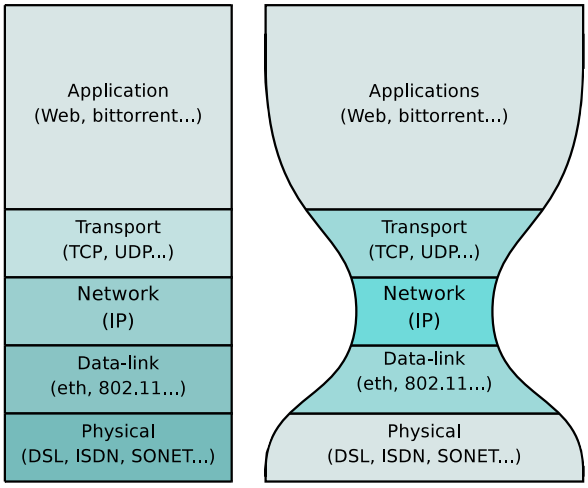
3.1 Historical Perspective

Paul Baran discussed distributed, electronic, store-and-forward networks and packet-switching in 1964, albeit under the different name of "message switching" [20]. His was speculative work for the US military, aiming to construct a decentralised, reliable communications network which would allow military communication between end-points to continue even in worst-case scenarios, via many redundant links.[1]

Donald Davies was initially unaware of Baran's work while pursuing similar ideas, though Davies' goal was to share scarce computing resources over networks, for which he also coined the term "packet switching" [21].

While Baran's military network was never built, these works provided a good basis for the design and construction of the ARPANET to commence in 1969, linking together

[1] This requirement for the work seems to be the source of the modern myth that the Internet is designed to withstand a nuclear attack.



(a) TCP/IP

(b) The "waist", indicating the common protocol hiding local networks beneath, allowing a multitude of applications above.

Figure 3.1: The Internet Protocol stack.

a handful of American universities. RFC 1 was published May 1969, intended to further discussion on the software to be hosted on ARPA interface message processors (IMPs) and the hosts connected to those IMPs. Further discussion of the design of the ARPANET can be found in [22, 7].

Vint Cerf and Bob Kahn introduced the idea of connecting two different networks via a network gateway, thus providing a mechanism to build networks of networks [23], using a common internetworking layer over the underlying networks. Gateways are designed to simply forward packets from one network to another, handling packet translation between different types of networks; the gateways themselves do not handle retransmission of lost data, making no assumptions about the reliability of an attached network greater than of a "best-effort" service.

The common internetworking layer later became the modern TCP/IP protocol suite, shown in Fig. 3.1(a). As initially described, TCP and IP were not separate layers in the network stack, but over time evolved into separate layers to allow for reliable and best-effort packet delivery with TCP/IP and UDP/IP respectively. This split emphasises the generality of the network functionality provided by IP, and is a good example of the end-to-end principle discussed in §2.2, where commonalities between different transport types are placed in the network infrastructure.

The ARPANET began a gradual shift to TCP/IP in mid-1982. Previously, the Network Control Program (NCP) had implemented the common network protocol for use between ARPANET nodes; on January 1st 1983, NCP was turned off completely, leaving IP as the primary means of communication across the network. Fig. 3.1 demonstrates the network layer as the common level of interaction, detaching the layers above from the network details beneath.

The version of IP deployed, IPv4, is still in widespread use today, and is defined in [24]. An IPv4 IP address is 32 bits wide, providing 2^{32} bits of address space, minus

various reserved addresses [25]. The address space was originally partitioned into three primary class sizes:

- Class A, with a network prefix of length 2^8, providing 2^{24} host addresses.
- Class B, with a network prefix of length 2^{16}, providing 2^{16} host addresses.
- Class C, with a network prefix of length 2^{24}, providing 2^8 host addresses.

This class-based addressing scheme quickly led to a fragmented address space: a class A address block provided more host addresses than most networks required, while class C address blocks provided so few host addresses that multiple blocks had to be purchased, or one (often wasteful) class B block had to be purchased instead. With so few class-B networks available, these addresses would have run out swiftly.

3.1.1 Developing Inter-domain Routing

The formalisation of the idea of a "network of networks" became a "network of autonomous systems (ASes)" with the exterior gateway protocol (EGP) as the common routing protocol between ASes to share routing state was introduced in October 1982 [26]. Motivation for developing EGP and ASes was to allow the network to continue scaling beyond its current state by creating a distinction between local network routing protocols and inter-domain routing protocols. The first inter-domain routing protocol was the Exterior Gateway Protocol (EGP).

The introduction of autonomous systems allowed for some flexibility with the network operator as to which routing protocols to deploy within their own administrative domain, provided they used EGP to communicate with neighbouring networks to provide the genera Internet service.

EGP did not aggregate addresses, and created a tree structure over which to route between autonomous systems. The successor to EGP, the Border Gateway Protocol (BGP) was introduced in 1989 [27], did not initially aggregate addresses, but did relax EGP's requirement of tree structured graphs.

Expansion of forwarding tables in the default-free zone of the BGP system (the DFZ, where routers generate forwarding tables encapsulating the whole of the address space without the use of any default routes) began to expand at exponential rates, due to the growth of the network, and from the many small address blocks which had been allocated. Classless inter-domain routing (CIDR) [28] was introduced from around 1994 to allow variable length network prefixes as a replacement for the class-based addressing system. As a solution to these problems, CIDR offers:

- Variable-length network prefixes replaced coarsely grained address classes with more finely grained classless address blocks of varying length, allowing for greater variation in the number of host addresses available to a network, thus offering a more efficient (i.e., potentially more tightly-packed) use of the total IPv4 address space, while also solving an artificial exhaustion of the address space.
- Aggregation of groups of smaller address blocks into one larger advertisement to be presented in the DFZ.

CIDR operates within the same 32-bit address space, and so only required minimal changes to existing infrastructure. The deployment of CIDR marks the last major change in IPv4 routing and addressing.

Beyond IPv4, the IETF IPng Area was formed in 1993 to look into a successor for IPv4, given the known problem of future IPv4 address exhaustion. It called for white papers detailing potential successors to IPv4 [29], ultimately recommending what was formalised as IPv6 in 1995 [30].[2]

3.2 Unicast Routing Protocols

Given the network structure of ASes, we have intra-domain routing and inter-domain routing protocols. I will cover the main protocols used to achieve both in this section.

3.2.1 Intra-domain Unicast Routing Protocols

There are two main types of routing algorithms for achieving unicast routing: link state and distance vector; there are various protocols which implement these algorithms which are in widespread use.

Distance vector algorithms operate by each node periodically sending a list of *destination:distance* pairs (the "distance vector") to each of its neighbours. On receipt of a distance vector, a node can check the distances announced by its neighbour against the currently known best distance, and choose whether to maintain the old route or use the new route.

Due to the time it may take to propagate reachability state across the network, convergence times demonstrated by distance-vector routing protocols can be slow. Thus, changes in network topology can lead to temporarily unreachable destinations, or routing loops.

Also, a simple distance vector protocol may suffer from the "count-to-infinity" problem, where a failed node will cease to advertise its distance vector to neighbouring nodes; in this situation, the other nodes running the routing protocol still have knowledge of the failed node, and continue to advertise it in their own distance vector announcements. These advertisements naturally become more attractive routes than over the failed link, and so routing loops begin to appear in the network. The count-to-infinity problem can be partially solved using the "split horizon" technique, where nodes will not advertise a route back to a neighbour if that neighbour is on the path to the destination. Split horizon with poison reverse allows for failed nodes to be removed from the routing system by advertising an unreachable distance to that node.

[2] Multiple white papers were received after the white paper solicitation by the IETF IPng working group [29], which went through various iterations before arriving at the final recommendation in RFC 1752 [30].

CNAT was only described as a "Work in Progress"; the Nimrod routing architecture [31] was considered "too much of a research project"; TUBA (TCP and UDP with Bigger Addresses) [32]; Simple Internet Protocol Plus (SIPP) [33] was a minor evolution from IPv4 which expands the available address space from 32 bits to 64 bits, and simplifies the IP packet header by removing (or making optional) rarely used IPv4 header fields; CATNIP (Common Architecture for the Internet) [34] was "too incomplete" to consider as a serious option.

TUBA and SIPP were both described as being workable solutions, but both would require modification. SIPP was chosen to be modified, and ultimately became IPv6.

The Routing Information Protocol (RIP) is a commonly used distance-vector routing protocol, which uses these split horizon with poison reverse techniques. The Interior Gateway Routing Protocol (IGRP) by Cisco is also a commonly used distance vector protocol using the same techniques to aid stability.

Distance vector protocols can take time to converge on a stable view of a network. As a network grows larger, the time taken to converge increases; this, coupled with the greater likelihood that network conditions will change more frequently in larger networks, suggests that distance vector protocols do not scale well relative to the size of the network.

In a link-state protocol, each node periodically broadcasts information on its incident edges (including the neighbouring nodes attached to those edges), allowing all nodes to quickly gather a full picture of the network. A shortest-path algorithm is run (e.g., Dijkstra's shortest path algorithm) at each node using this information, allowing each to compute the shortest path from itself to all other known destinations, thus generating a forwarding table for all incoming packets.

A link-state protocol allows for faster convergence than a distance vector protocol, but periodic broadcasting of link states between nodes becomes undesirable as the number of routers required to service a network grows.

The most commonly used link-state protocol currently in use is Open Shortest Path First (OSPF) [35]. Another widely deployed routing protocol is Intermediate System to Intermediate System (IS-IS) [36]. IS-IS runs alongside IP, rather than on top of it like OSPF; it was designed as a general routing protocol, whereas OSPF was designed with IP in mind. To lessen the burden of flooding link state across an entire network, both protocols allow for two levels of hierarchy to permit the clustering of groups of nodes together.

Intra domain routing algorithms are well understood, with protocols often widely deployed and used to maintain unicast routing in networks of various sizes. These algorithms imply that routers maintain a full picture of the network.

3.2.2 Inter-domain Unicast Routing Protocols

Where intra-domain routing protocols forward packets between routers within the same autonomous system (i.e., conceptually, on the same network), inter-domain routing forwards packets between these ASes to reach their ultimate destination. The level of granularity that an inter-domain routing protocol can choose to deal with is that of the autonomous system as an aggregation of routers, rather handle the individual routers themselves.

BGP-4 is the routing protocol currently operating between autonomous systems on the Internet. BGP is a path vector algorithm, a variant on distance vector protocols which maintains paths to endpoints in routing table entries, which enables the avoidance of the well-known "count-to-infinity" problem. The currently deployed version of BGP is BGP-4 [37], which implements CIDR [28] for address aggregation.

Control over policies can be used to reflect business relationships between ASes, allowing for one AS to prefer certain routes over others, or filtering routes to influence the behaviour of other ASes. Given this policy-influenced graph, BGP attempts to route

across the shortest AS-path through the fewest autonomous system as possible, given the various policy rules within that AS.

Within an AS, there may be multiple BGP routers maintaining connectivity with other ASes; these BGP routers form a mesh over which they exchange forwarding information to allow correct packet forwarding. A distinction can be made here between IBGP (for internal BGP connections) and EBGP (for external BGP connections).

There are three common relationship types between ASes: customer-provider relationships, where an AS pays another AS for connectivity; peer-to-peer relationships, where ASes exchange traffic where it is beneficial to avoid using the upstream provider; and backup relationships, used when other routes have failed. ASes can exert some influence in which ASes incoming packets are routed to due to the nature of longest-prefix matching in CIDR: if a specific route is preferred for a subset of the address space owned by an AS, then that AS can advertise that smaller address block alongside its other advertisements.

3.2.3 Anycast Routing

Anycast routing is essentially a special case of unicast routing where one address may resolve to multiple endpoints, but no more than one endpoint receives a packet sent that address. It provides a natural mechanism to support load balancing and redundancy of servers.

Anycast is achieved in by injecting the same address into the routing system from multiple points, and allowing the multiple sources to propagate across the network. The routing protocols employed should forward packets to the "nearest" destination.

Anycast services deployed in the Internet are rare, however; none of the IPv4 address space is dedicated to offering anycast services. Normal unicast addresses must be used to advertise an anycast service, but ASes are generally unwilling to carry address block advertisements with prefixes longer than a 24 bits, so a "/24" address block has to be purchased to provide a service reachable via anycast, leading to wasted addresses. DNS rootservers are one example of a service which uses anycast [38].

3.3 Multicast Routing Protocols

Widely deployed local-area broadcast networks such as Ethernet offer a natural mechanism for multicasting data from one node to many: to reach all nodes, no more Ethernet frames need be transmitted than for a unicast operation, with hardware in network interfaces filtering frames not intended for that host.

Extending multicast into a wide-area inter-network environment, where networks are linked together using network-layer bridges, requires construction of distribution trees to avoid mere flooding of packets to all nodes.

Steve Deering approached the problem in 1988 [39, 40], proposing mechanisms to provide multicast on extended LANs (LANs connected by link-layer bridges) and to provide multicast across independent networks (networks connected by network-layer bridges). The discussion of the former focusses on link-layer bridges forwarding based

on LAN addresses (e.g., MAC addresses). The discussion of the latter focusses on network-layer bridges forwarding based on IDs (e.g., IP addresses) globally unique across the inter-network (though LAN addresses need not be unique). This work led to the formalisation of DVMRP and MOSPF, both described in §3.3.2 for providing multicast in the internetwork environment.

The adopted IP multicast service model, as defined in Deering's work and referred to here as the Deering multicast model, attempts to adhere to the connectionless, datagram-based end-to-end principle embraced by unicast UDP, where the network is viewed as a best-effort datagram service and the end-hosts handle more complex functionality such as connections, packet ordering, etc. This service model has driven most research into IP multicast since its definition. The model operates as follows:

- A host must announce interest in receiving data from a group, leaving the network to set up the state required to forward data to receivers. Deering essentially designed the local-area group membership announcement mechanism, where endpoints periodically announce which groups they are a member of: formalised as IGMPv3 in IPv4; Multicast Listener Discovery (MLD) v2 in IPv6.
- Any host may send data to a group; a host does not have to be a member of a group to send packets into it.
- Routers conspire to deliver a source's data to all interested receivers.

This service model defines an "any source" system, where any node on the network may be a source and all listeners will receive data sent by sources, assumes that each host in a group is a potential source. However, the service model does not dictate how to route from a source to destinations. The distribution trees which may be generated to allow multicast data dissemination are either:

- *Shortest Path Trees*: Where a tree is generated from each active source to all group members.
- *Shared Trees*: A single shared tree between all nodes, which each node uses to send data. A generated tree is, ideally, a minimal spanning tree between all nodes approximating a Steiner Tree.

The various protocols implementing these schemes are covered later in §3.3.2.

Initial deployment of the IP Multicast service was encouraged through the use of the MBone, an overlay network employing tunnels to connect multicast-enabled "islands". Tunnels for the MBone required manual reconfiguration, and more packets would have been duplicated than in a totally native environment [41].

3.3.1 Address Allocation for Inter-Domain Multicast

Multicast addresses form a flat namespace, without the hierarchical assignment used by unicast addressing. Within the IPv4 address range, IP Multicast uses addresses in the range 224.0.0.0 – 239.255.255.255, formerly Class D addresses prior to the introduction of CIDR.

Certain subsets within this address space are managed by the Internet Assigned Numbers Authority (IANA):

- 224.0.0.0/8: The "Multicast swamp", in which some addresses have been assigned arbitrarily.
- 224.0.0.0/24: Used for local networks, low-level topology discovery, etc. Forwarded within a private realm only.
- 232.0.0.0/8: Source-Specific Multicast (PIM-SSM; see §3.3.3).
- 233.0.0.0/8: The GLOP address space, where the two middle octets in the address are filled with an AS number, giving each AS dedicated space for 256 multicast groups [42].
- 239.0.0.0/8: Administratively scoped addresses [43], routed only within an administrative domain. Analogous to private IPv4 addressing blocks..

Aside from these pre-defined ranges, the multicast address space is reserved by IANA [44], which applications must not use.

3.3.2 Intra-Domain Multicast: Any-Source

Based on Deering's multicast model, most multicast protocols which have been defined are any-source multicast protocols. There are two broad categories for multicast groups in an internetworked environment: *dense* groups, and *sparse* groups. The former are groups where the number of group members is close to the total number of nodes in the network, and the latter are groups with few members relative to the size of the network (note that sparse groups on an Internet scale are therefore not necessarily *small* groups). These prompt two main classes of multicast algorithms: dense-mode algorithms where the network assumes all nodes are members unless a node rejects packets from a group and sparse-mode algorithms where the network assumes no nodes are members of a group until explicitly joined.

Dense mode multicast systems are soft-state protocols and driven by the presence of data passing from source to group; sources which do not continuously contribute to the group can initiate a round of "flood & prune" each time they send new data into the network. To fit the terminology, pruned branches will grow back.

The Distance Vector Multicast Routing Protocol (DVMRP) is essentially multicast RIP [45]. DVMRP employs a "flood and prune" mechanism: a packet sent to a multicast group is broadcast to the network by routers on the shortest path back to the source, and routers send prune messages back in the direction they received multicast packets if they currently serve no members of that group. Branches not on the shortest path to a receiver are pruned if multiple routers can serve the same receiver. If a new member appears, a router can graft a link back onto the pruned tree; otherwise, pruned branches will periodically grow back, and the flood/prune process starts over.

DVMRP is, therefore, a data-driven protocol: in the absence of data from a source, the distribution tree from that source will time out. It is the existence of data packets which prompts the building of the tree. DVMRP creates reverse-shortest path trees.

Protocol Independent Multicast - Dense Mode (PIM-DM) [46] is similar to DVMRP, except that it makes does not rely on any particular topology discovery mechanism.

This simplifies PIM-DM when compared to DVMRP, but may incur more duplicated packets prior to the "prune" operation.

Multicast Open Shortest Path First (MOSPF) is a link-state multicasting protocol [47]. Changes in group state are flooded throughout the network just as other link-state advertisements, allowing all routers to store a map of the network. On receipt of a multicast packet from a new source, a router will compute the shortest path spanning tree from the source to all known members of that group, and cache the result to use against future packets from the source; thus, there is a little additional computation on the first packet in an MOSPF environment, but future packets should be routed as quickly as in DVMRP. MOSPF therefore removes the need to flood data packets throughout the network, but incurs the additional overhead of flooding network state between all routers.

Dense-mode multicast protocols make an assumption that receivers for a multicast group are common throughout the network. Based on this assumption, data packets are forwarded throughout the network, the onus being on routers to maintain enough state to forward packets sent to a group only along links with known group members. Further, all routers, even those not on the distribution tree for a source, must retain per-source state.

Dense-mode protocols either require too much state or too much wasted bandwidth to scale to situations where the receiver set is sparsely distributed across the network. If we consider the Internet, even for a group where we might expect high numbers of receivers (such as a TV 'broadcast'), the size of the receiver set will be significantly smaller than the number of nodes on the whole network.

Sparse-mode protocols were designed to alleviate the state/bandwidth burden of dense-mode protocols. The key distinction between sparse and dense mode protocols is that in sparse mode protocols, only those nodes on the data path from source to destinations maintain state relating to the source or the group. They employ some form of rendezvous point to aid in the setting up of this state. These algorithms generally create reverse shortest-path trees.

Core-Based Trees (CBT) [48] are one mechanism to address scaling problems inherent in dense-mode protocols: Core-based trees centre around a *core*, from which a shared tree between all nodes is created; data to be sent to the group is first sent to the core. CBT therefore requires less storage per router, since forwarding can be done on a per-group basis rather than a per-source basis. CBT uses explicit join and leave messages to setup and destroy forwarding state, and hop-by-hop mechanisms to achieve reliable delivery of control messages. OCBT is a provably correct version of CBT which will not partition trees should a node leave or fail, [49]. Multiple cores are considered for use with (O)CBT to improve scalability, but are deemed too complex; a key concern in an (O)CBT system is therefore traffic concentration on the core of the group. (O)CBT is not widely deployed.

PIM-SM [50] uses existing unicast routing tables to forward control packets toward rendezvous point; if unicast routing is asymmetric, the resulting tree is a reverse shortest-path tree. Unlike (O)CBT's hard-state, PIM-SM employs a soft-state periodic refresh for state, which will time out if not refreshed. Control messages are piggybacked on data packets, since the IP Multicast service model allows any sender to send into a group without first asking for membership.

PIM-SM initially builds a shared tree by sending join messages toward a rendezvous

point (RP) (analogous to an (O)CBT core). Once data is flowing, a receiver can leave the shared tree and generate a shortest path from the source, a mechanism which involves an intermediate on-path router (which knows the data it is forwarding to the receiver is not being routed via the shortest path, given that it is using the unicast forwarding tables) sending a join message toward the source, then pruning the previous incoming route. RP location is achieved via an out-of-band mechanism. Generating shortest paths from the shared tree allows a group to shift away from traffic centred around the RP, and could therefore avoid the traffic concentration problem, albeit at the expense of much added complexity to the protocol. A variant on PIM-SM is Bidirectional PIM, which generates bidirectional shared trees rather than unidirectional shared trees rooted at the RP [51].

Both PIM-SM and (O)CBT require on-path state. This, arguably, partially violates the fate-sharing argument outlined in §2.2, in the sense that applications should not have to rely on complex state replication to maintain an application-level association. Without said replication, a portion of the group membership will be disconnected from the group if one on-path router fails.

3.3.3 Intra-Domain Multicast: Source-Specific

As suggested in §1.1, there exist many applications which best suit a restricted multicast model of delivery, offering "send" privileges to just one source, with the rest of the group as passive listeners. This is obviously different to the 'any-source' model of communication proposed by Deering, used for much of the subsequent research and protocol development. A source-specific multicast model caters for a different set of applications from those applicable to an 'any-source' model.

EXPRESS (EXPlicitely REquested Single Source) Multicast [52] evolved directly into PIM-SSM (PIM-Source-Specific Multicast) [53], and offers multicast groups with one source. A multicast channel is defined by the source's IP address and a destination address taken from the IP Multicast address pool reserved for SSM use. In this way, each source has the potential to be the source of 2^{24} groups, expanding the number of potential groups considerably, and allowing for a much more efficient use of the multicast address space.

SSM terminology is slightly different to standard multicast terminology: each source is in charge of various *channels*, not groups, and listeners subscribe and unsubscribe to channels rather than join and leave.

3.3.4 Inter-domain Multicast Routing

Inter-domain multicast has not been widely deployed. There exist a set of protocols [54] which together aim to provide an inter-domain multicast service. These protocols are described here.

BGMP, the Border Gateway Multicast Protocol [55], generates a centre-based bidirectional tree between ASes; the AS who owns the group address, possibly claimed using a mechanism such as the Multicast Address-Set Claim protocol (MASC, [56]), becomes the root, or centre, of that group. BGMP has not been deployed, however.

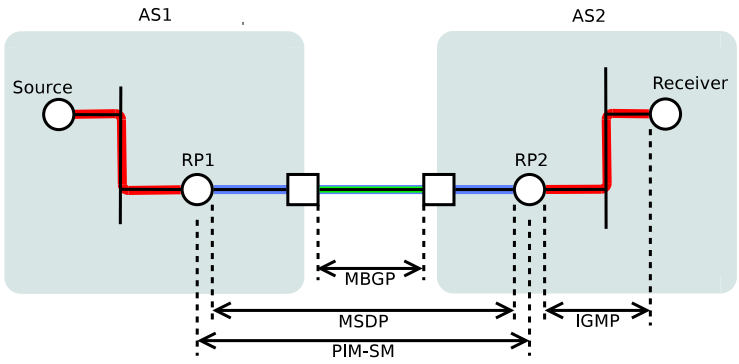


Figure 3.2: Illustration of the commonly used protocols for interdomain IP Multicast.

MBGP, Multiprotocol Extensions to BGP, is widely deployed, and is used to maintain the distance to source addresses such that reverse path forwarding calculations can take place at routers [57]. Differing policies for unicast and multicast traffic may then affect the routes chosen for each. MBGP is essentially used to create a topology over which PIM-SM or PIM-SSM can operate.

MSDP, the multicast source discovery protocol, connects PIM-SM rendezvous points between ASes [58]; rendezvous points send (flood) to each other source active (SA) messages to announce the arrival of a new active group in the multicast address space. Flooding is required to announce to other RPs that new groups have been advertised quickly, and new groups must be advertised quickly in the absence of a centralised authority issuing multicast addresses. MSDP, however, is a short-term solution which cannot realistically scale to an Internet-scale universally utilised protocol. As such, MSDP is IPv4 only, and is not to be implemented for IPv6. Embedded RP, specified in RFC 3956 is the IPv6 solution, in which the location of the rendezvous point is embedded in the IPv6 group address.

Over the topologies created by MBGP and MSDP, PIM-SM and PIM-SM are commonly used to generate the actual data distribution trees required for packet forwarding. Figure 3.2 offers a simplified overview of where these protocols are actually used in an inter-domain setting.

3.4 Discussion of Shortcomings in the Current Architecture

The current Internet architecure, as described in the preceding sections, is based on technical decisions a quarter of a century old. Application use has changed significantly since then. The primary goal of supporting unicast traffic stems from the resource sharing goal of the initial work in the 1960's and early 1970's. However, as suggested in §1.1, users can expect many types of group applications to be used across the modern Internet.

If we consider the two broad classes of addresses in use today: one-to-one addresses (unicast and anycast), and one-to-many addresses (multicast and broadcast), we can

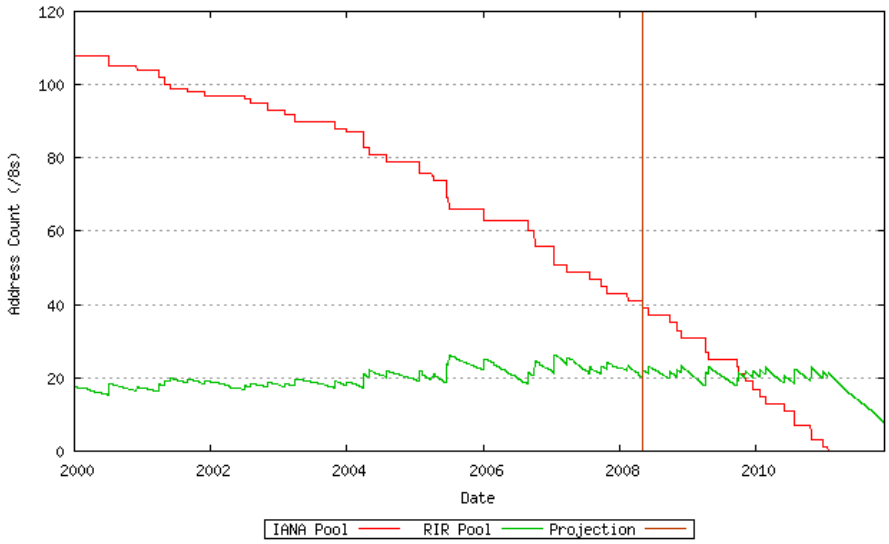


Figure 3.3: IPv4 address shortage: IANA and RIRs; from [1].

see a distinction in the semantics of addresses applied to each of these classes:

- In the former class, the address is intended as a network locator, containing enough information for a packet to be forwarded from router to router to its final destination.
- In the latter class, however, the multiple distinct destinations for a packet implies that this address alone is not enough to locate the endpoints: additional state is required to allow the packet to arrive at all its destinations, be it additional router state, additional information carried within the packet header.

The distinction is clear in terms of how the routing infrastructure uses the different classes of addresses: it is not possible in the current Internet architecture to initiate communication with one other host and subsequently admit a third party to the communication; unicast routing protocols do not handle this, and so unicast transport protocols do not support it.

In this section, I will cover some of the various shortcomings evident in the Internet architecture currently deployed: the state of the IPv4 addressing scheme (§3.4.1), the scalability of inter-domain unicast (§3.4.2), the scalability of inter-domain multicast (§3.4.3), and mobility (§3.4.1).

3.4.1 IPv4 Addressing

Not only is there a distinction between the semantics and usage of IP addresses, there is an immediate problem in terms of a shortage of addresses: most of the address space has already been claimed, and is due to run out completely within the next few years (Fig.3.3).

Endpoint Identification and Network Location

Despite the layering designed into the TCP/IP protocol stack, a lack of mobile nodes at the time allowed the designers to make the simplifying assumption that an endpoint identifier had the same lifetime as a network locator, and so there was no need to differentiate the two between the network and transport layers; to do so would have introduced unnecessary complexity where it was not then needed.

As suggested in §2.3, the tight coupling of the two layers means that it is difficult for the network layer to evolve independently of the transport layer. So, an IP address is currently used both as a network locator (at the network layer) and as an endpoint identifier (at the transport layer).

IPv6 was proposed as a minor, but incompatible, modification to IPv4, which provided a simplified packet header to ease packet processing, and also a much larger address space. IPv6, however, did not decouple the use of the IP address between layers, and so suffers from the same scaling inefficiencies apparent in the IPv4 routing system, which I'll cover in §3.4.2.

Network Address Translation (NAT)

A NAT is a packet relay connecting two different addressing realms thus destroying the original goal of unique IP addresses identifying endpoints. The existence of NAT causes problems for end-to-end protocols such as TCP. NAT has come about for two reasons: the growing lack of available IPv4 addresses, and the apparent expense in purchasing larger address blocks.

NAT allows an organisation to handle its own private IP address space and its public-facing address space. NATs have arguably extended the lifetime of the IPv4 address range also. However, in rendering many hosts unaddressable from the public Internet, they also hinder the operation of a whole breed of potential peer-to-peer applications. File-sharing is the commonly thought-of peer-to-peer application, but other applications suffer: VoIP is a useful example.

One of the aims of IPv6 was to remove the need for NAT, which it aims to achieve by introducing the huge address space capable addressing up to 2^{128} entities, a large enough address space that NAT should not be required. The rationale behind the design of IPv6 was to restore the original end-to-end nature of the architecture.

An argument could be made *for* NAT, in that large networks may wish to "hide" parts of their network in the private address space, protecting them from some forms of malicious attack from outside the organisation; NATs may only provide the *perception* of additional security, however.[3]

[3]Though there are protocols being developed which aim to work around them: STUN can be used by hosts behind a NAT to discover the public IP address and port number created for an outgoing connection; TURN can be used to work around especially restrictive NAT configurations by using a relay for data communication; and ICE is a protocol to test multiple possible connection strategies and choose the best one on a per-connection basis. NAT behaviour can vary considerably due to a lack of standardisation, and so traversal is difficult.

Mobility

The original TCP/IP proposal made the (then valid) assumption that nodes were stationary. Now, however, we are seeing a more dynamic network partially moving toward mobile connectivity, in various scales:

- Mobile devices, including mobile phones, PDAs, laptops, etc, can move between wireless access points and/or suitable wired locations (e.g., office and home).
- Vehicles can move from one location to another: car drivers commute; trains might connect to two different networks in two different cities; planes might connect to two different networks in two different countries.
- Changing ISP, can also be considered as a "slow" form of mobility.

Each of these operations might imply assigning a new set of IP addresses to the nodes who have changed location. The overlap of using an IP address as a locator for the network layer, and as an identifier for the transport layer, requires that transport layer identifiers are modified when an IP address is reassigned. Achieving mobility is difficult, especially with the tightly-coupled transport and network layers in the protocol stack. Mobile solutions involve electing a "home" IP address, which acts as a relay for a mobile node; this may introduce triangular routing, however, where outbound packets from a mobile device can be routed directly to their destination, but inbound packets must be routed via the relay.

3.4.2 Scaling Inter-Domain Unicast Routing

Available data suggests that the Internet graph is a "scale-free" graph, i.e., a graph demonstrating a heavy-tail in the node degree distribution. The Internet graph also contains many triangular subgraphs, confirming that the Internet graph is not hierarchical.

Forwarding tables appear to be expanding at a super-linear (possibly exponential) rate [59, 2] (Fig. 3.4). Routers must be able to look up their forwarding table and decide on the output port for a packet within a short timeframe, and larger forwarding tables require longer lookup times, and so routers become more difficult (and expensive) to design and maintain. If a router cannot handle packets at line-speed, then it becomes a bottleneck (rather than a slow link being the bottleneck). Routers with sufficient fast memory to hold the full BGP table found in the DFZ which can route at line rate will therefore become more expensive, and require more power to fulfil their task [60]. Likewise, network managers must ensure that their routers contain enough memory to be useful for a lifetime which matches their cost.

The root of the problem of super-linear growth of forwarding tables, aside from natural continued growth of the network, appears to be:

- Multihoming, where end-sites choose to purchase transit service from more than one service provider, usually to boost connectivity.

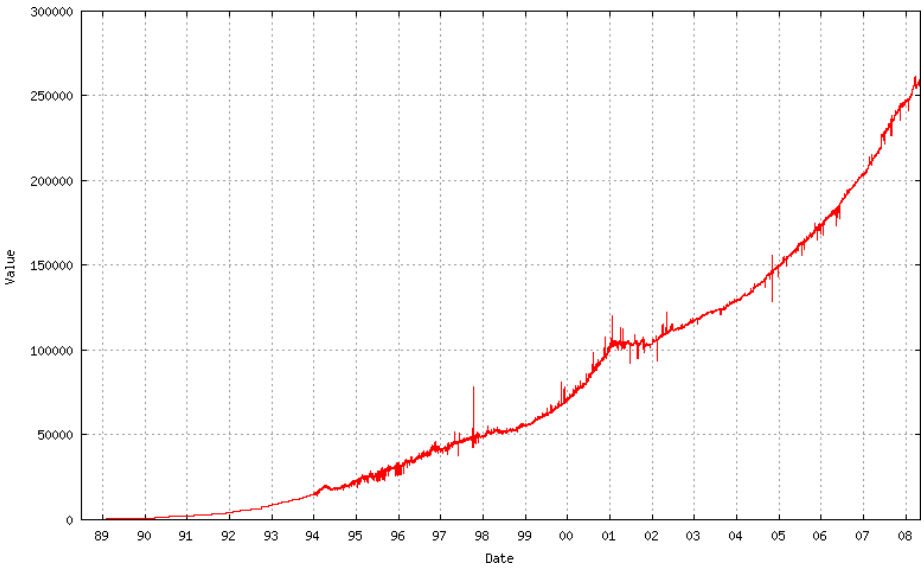


Figure 3.4: BGP FIB growth in the DFZ; from [2].

- Network address translators (NATs) allow end-sites to build large networks, with only a small subset publically addressable from the Internet, allowing organisations to purchase smaller address blocks [59]. NAT, therefore, encourages the use of smaller address blocks which, if bought from PI space, cannot be aggregated.
- The identifier/locator overlap inherent in the TCP/IP stack leads to the introduction end-sites looking to buy address space which does not adhere to the hierarchical principles CIDR attempts to leverage. (Provider Independent, PI, addressing.)

The problem is exacerbated by current IP address allocation strategies [61], the nature of inter-AS business politics [62], and the nature of the Internet-graph [63].

Multihoming and PI Addressing

BGP allows non-hierarchical structures in the Internet graph to accomodate business interactions between service providers (peering arrangements, to share traffic load between ASes and thus saving on volume of transit to be purchased from an upstream provider) [64].

Thus, some of the current scaling inefficiencies come from the CIDR mechanism which attempts to impose a hierarchical naming scheme intended to reduce the size of forwarding tables over a graph which is not hierarchical.

Much of the expansion in the forwarding tables in the DFZ can be attributed to end-sites opting to multi-home. Such an arrangement provides the benefit to the end site that they are no longer reliant on one ISP to make strong service guarantees; instead, they opt for redundancy with more than one route in and out of their network. To achieve this redundancy, each route must be advertised back out into the BGP routing

system, suggesting that the advertised address block cannot be aggregated by at least one service provider.

The most common way multi-homed end-sites are addressed, however, is via provider-independent (PI) address blocks, as opposed to aggregatable provider-assigned (PA) addressed blocks. Provider independent addresses make up a substantial number of address allocations by regional Internet registries (RIRs). As none of these allocations can be aggregated, these relatively small blocks must be advertised independently of all other addresses in the DFZ.

The uptake of provider independent address blocks reflects a shift in customer demands:

- Renumbering a site when changing provider is difficult, and end-sites do not wish to take on the hassle of renumbering to adhere to the requirements of the routing infrastructure on such a change. A PI address block can be used by the end-site no matter the provider.
- Customers consider an IP address block to be something akin to a telephone number, i.e., as something which uniquely identifies their site. End-sites wish to retain control of IP address blocks.
- Multihoming destroys the topology-based prefix-aggregation required to gain benefit from CIDR; for an AS to be reachable through any of its connections, the global routing system must be aware of both routes, and so prefix must be injected into the routing system via both parent ASes. If drawn from PI address space, this cannot be aggregated by either provider.

If the transport and network layers were decoupled, then the transport layer would provide one identifier to the applications, the network layer would be assigned multiple locators (hierarchically assigned from each provider), with the transport able to choose which of the network locators to use, thus achieving the two aims of greater reliability and unique identity.

Other General BGP Trends

Aside from the scalability issues introduced by multihoming, there are various other trends which affect the size of forwarding tables, [59]:

- BGP communication overhead is known to be exponential relative to the size of the network.
- The span of originating addresses per AS is getting smaller. (i.e., ASes are advertising smaller blocks in general.)
- Despite the number of ASes constantly increasing, the average AS path length has essentially remained constant at around 3.5 AS hops. To achieve this, the interconnection degree between ASes is getting higher.

Kleinrock and Kamoun's 1977 paper on hierarchical routing explored the relationship between forwarding table sizes in a hierarchical network and the distance between

nodes in that network, suggesting that depths of hierarchies must grow quickly with respect to the number of nodes present [65] to maintain reasonable sizes for forwarding tables.

Thus, CIDR as a solution to the scaling properties of the DFZ forwarding table, employing hierarchical address assignment to encourage address aggregation in the centre of the network, is failing in part due to the hierarchy in the network not increasing with the number of networks and nodes.

CIDR, given that it aggregates address blocks, is also at odds with the possibility of offering node mobility directly in the inter-domain routing infrastructure: individual IP addresses cannot be advertised in the BGP system, because ASes will not carry adverts this small. However, aggregation is often defeated with PI address blocks, and by ASes deliberately deaggregating blocks to aid traffic engineering.

Clearly, the currently deployed inter-domain routing system has served its purpose well, but it does not seem capable of scaling far beyond what it is managing currently.

3.4.3 Scaling Inter-Domain Multicast Routing

Deployment of Deering's multicast model has been slow, arguably for multiple reasons:

- State explosion: Existing protocols may require, at best, on-path state, and so there are concerns over cost of hardware and large-scale feasibility if multicast as a service were to take off.
- Related to this, there is a lack of deployed, scalable inter-domain routing protocols.
- Security: The Deering multicast model is open, allowing any node to request packets from any group (within certain limitations, §3.3.1).
- Denial of service: Packets can be sent into any group, with the network replicating packets toward all receivers. In other words, packet replication may be high, delivering malicious packets to many hosts at little cost to the attacker. In this sense, the model attempts to mirror the unicast model where hosts may inject packets into the network with any destination without any prior setup.
- The initial assumption of an ASM model led to the development and deployment of ASM-based protocols first, with the SSM model following much later.
- Accountability: In a multicast system, an AS other than the originating AS may bear most of the costs for packet duplication and delivery.

The slow deployment of Multicast has led to the creation of various end-host, peer-to-peer solutions, which will be covered in §4.3.1.

The various inter-domain protocols were discussed in §3.3.4. Of these, MBGP, MSDP, and PIM-SM are the current "best" solution to offering native IPv4 multicast. Yet MSDP cannot scale to offer a global multicast service, given that it floods source activity between RPs in connected ASes. The other major concern is that of on-path state to offer PIM-SM. These solutions require a total deployment from source to destination to enable the multicast service.

Multicast Address assignment

In the IP multicast addressing model, the aggregation benefit of hierarchically assigned unicast addresses is lost, as members of a group may be distributed over a large geographic area (though there has been some research into the aggregatability of multicast addressing, [66]). In essence, IP Unicast and IP Multicast share the same address space but are used very differently at the network layer.

Multicast address assignment is handled by IANA, though much of the space is still marked as "reserved" (§3.3.1). With a flat address space, solutions such as MSDP [58] or MASC [56] to manage multicast address ownership are essentially a side-effect. The flat address space makes address ownership and assignment difficult.

GLOP addressing partially fixes this; rather than present a totally flat address space, GLOP provides each AS with 256 multicast groups; the existence of the AS number in each multicast group address allows for a clearer sense of group ownership, and removes the issue of advertising active groups (as in MSDP) [67].

3.4.4 Conclusion

In summary, the current Internet architecture exhibits problems in terms of being able to offer both unicast and multicast service in the near future. These problems may be overcome in an ad-hoc manner over time. Ad-hoc solutions, however, may make the architecture more complex, while the success of the architecture to date has been its simplicity. In this light, long-term, "blue-skies" attempts to re-design the Internet architecture are as worthwhile of our attention as the ad-hoc solutions to the problems presented here.

Chapter 4

Alternative Architectures

The shortcomings of the existing architecture have prompted a variety of works which aim to solve these problems directly. There are various works which aim to offer a clear identifier/locator split within the current architecture. These works often also attempt to solve the scaling problem observed in the forwarding tables of the DFZ. There are various other "clean slate" designs for network architectures, which explore other concepts such as mobility, network heterogeneity, etc.

The emphasis in many of these works is that there may be more to be gained from clean slate designs rather than by making minor tweaks to the existing system. Many problems inherent in the current Internet architecture have stemmed from homogeneity in protocols and difficulty in being able to evolve [15, 62, 68].

In this chapter, I will cover the broad range of work which either attempts to fix a problem in the existing architecture, or attempts to discuss new ideas applicable to network architectures. In §4.1 I will cover various alternative approaches to implementing general multicast on the Internet without a full deployment of the service. In §4.2, I will discuss attempts to split apart endpoint identifiers and network locators in the current Internet architecture. I will look at alternative routing mechanisms, including some explored by network overlays and overlay multicast, in §4.3. I cover compact routing, an interesting theoretical area which aims to place mathematical bounds on forwarding table sizes, in §4.4. I'll look at other architectures, or otherwise interesting ideas, in §4.6, before rounding off this chapter in §4.7 with some discussion on these various ideas presented.

4.1 Alternative Approaches to Offering Multicast

Despite partial deployment of the key protocols required to provide multicast in an inter-domain setting, deployment has been slow. There have been mechanisms proposed, however, to encourage eventual widespread deployment within the existing architecture, and various other application-level mechanisms designed as methods of offering a multicast service to applications without native network support.

'Automatic IP Multicast without explicit Tunnels' (AMT), has been proposed as one

possible solution to encourage further deployment of existing multicast proposals [18].[1] AMT permits non-multicast sites to receive packets from SSM and ASM groups and, if not located behind a NAT, permits the sending of data into an SSM group also. AMT *relays* would be located in the public Internet, and addressable via a well-known anycast address, allowing an AMT *gateway* (possibly the host itself) to register to join a group via the relay. Multicast data is then unicast from relay to gateway. In theory, this would provide an incentive for deploying multicast (or, more accurately, a disincentive for *not* providing native multicast) in that, if this were to gain traction, ASes which do not enable multicast would ultimately have to carry much more unicast traffic. Similar ideas are presented in [69], which aims to link up multicast islands using automatically constructed unicast tunnels.

Free-Riding Multicast (FRM) attempts to provide the multicast model with only minor modifications to BGP, potentially allowing for an easier transition to a multicast-enabled Internet [70]. FRM proposes that active group announcements are carried in normal BGP traffic, with group addresses mapped against these AS numbers which have active participants. The border router in the source AS for a packet calculates the distribution tree using this mapping of recipient ASes for a given group number, and encodes the links on this tree into a fixed-size (e.g., 100 byte) packet header, such that downstream ASes with a different view of the network can forward down correct branches. It's possible in this scheme to introduce false positives, whereby a mechanism similar to pruning is used to stem an incorrect flow. FRM multicast seems to perform well with respect to an idealised IP Multicast system, but the additional overhead per-packet might be rather large for some applications, and additional state required per AS to hold information on active groups may require more expensive hardware per AS.

The most commonly used multicast tree-generation protocols are PIM-SM and PIM-SSM, which require on-path state between source and recipients; while AMT above attempts to work around this problem by automatically tunnelling through parts of the network not multicast-enabled, the "preferred" option is native support for PIM-SM/PIM-SSM, and the on-path state those protocols entail. One potential vector to multicast deployment is to lessen the state burden of these protocols.

REUNITE (REcursive UNIcast TreE) is an interesting multicast scheme in that it doesn't use the class D address space [71]; instead REUNITE uses unicast addresses to forward packets to their destination, and intermediate routers forward REUNITE packets toward their next branching node without the need for tunnelling or any modification to intermediate routers. A REUNITE router holds two tables: a Multicast Control Table (MCT) and a Multicast Forwarding Table (MFT); branching nodes in REUNITE each store a subset of the receiver list in a forwarding table indexed by *(source node address; port number; destination address(es))*, and create new packets for each of the additional destination addresses listed in the entry. In REUNITE, multicast forwarding state need only be stored at branching nodes, and thus reduces state requirements considerably when compared to PIM-SM and (O)CBT. The MCT allows a router to know when it should become a forwarding node which duplicates packets, but unless the router is required to duplicate a packet, it introduces no additional forwarding state. Since REUNITE uses unicast tables to forward packets to receivers, data delivery toward receivers might follow a different paths to control packets from receiver to source.

[1] A more general version of this for deployment of new versions of IP is presented in [17].

HBH (Hop-By-Hop Multicast routing protocol) is a very similar scheme to REUNITE, but it does use the class D address space, in a manner similar to PIM-SSM [72]. MFT trees in an HBH distribution tree need only index by *(source address; group address)*, and do not need to use the receiver address; HBH explicitly forwards toward HBH router locations, where REUNITE only uses end-host locators, with those routers then performing a lookup against the *(source address; group address)* held in the packet, to find the next hop(s) for the packet. By using native IP Multicast addressing, HBH has the advantage that a native multicast cloud (i.e., multicast on a LAN) can be a leaf of an HBH distribution tree. In the same way as REUNITE, HBH only requires additional forwarding state at branching points.

Xcast is a simpler scheme, which removes the need for group addressing altogether, by simply listing multiple destinations in each packet header [73]; routers lookup against each destination, and forward packets with accordingly restructured Xcast headers. This is a solution which clearly does not scale to large groups, but may be useful for small, short-lived sessions.

None of these mechanisms have, as yet, been widely deployed. Another method of providing multicast functionality in the Internet has been the introduction of application-level multicast. Various end-system multicast systems have been explored, which aim to offer multicast to often ad-hoc groups, such as conference calls. Given the lack of deployment of native IP multicast, these network have been built to explore providing general multicast in an ad-hoc, end-host, per-group manner instead. Various similar projects have been published.

Narada constructs a mesh between members of the group, from which spanning trees are created for data distribution, with each tree rooted at a source within the group [74]. The Narada overlay protocol probes potential links between peers to provide an incremental optimisation of the mesh used to create distribution trees. DVMRP can be run across the generated mesh to create the spanning trees. Orta is based on the Narada work, but replaces the distance vector approach with a link-state protocol instead, such that changes to the mesh topology are distributed throughout a group quickly [75]; this allows Orta to be better suited for real-time communications applications.

Other network overlays aiming to provide a general multicast service include: ALMI, which builds a minimum spanning tree (MST) between all members of the multicast group to carry multicast data [76]. Each node has a connection to a session controller node, which is responsible for organising tree structure based on information returned by group members, and deals with members joining and leaving. FatNemo uses the IP network beneath to its advantage by organising end-hosts in the tree such that higher-bandwidth nodes are closer to a tree's root [77] (creating a "fat" tree). HyperCast forms a hypercube from group members, group members becoming vertices in the hypercube [78]. Spanning trees are embedded into the hypercube easily, while control traffic is transmitted along the edges of the hypercube. Scattercast [79] aims to provide a large, semi-permanent multicast infrastructure which uses SCXs (ScatterCast proXies) at known locations; these SCXs are application-level software components which form an overlay network [79]. Clients can then connect to SCXs using IP Multicast if it's locally available, or by using normal unicast links if not.

These protocols require that a considerable amount of control traffic is sent regularly between peers to ensure connectivity between nodes and improve overlay performance. Without native network support, these protocols can never perform as well as native

multicast, both in terms of latency between endpoints (especially if the overlay is not topologically aware) and in terms of the number of duplicated packets.

These overlay networks are susceptible to failure, given the reliance on end-hosts. In this sense, the principle of fate-sharing is violated here in that the failure of an endpoint may affect a subset of the group without the application being at fault; the peer-to-peer overlay is essentially mimicking multicast functionality which may be best offered in the network.

In summary, the lack of a pervasive multicast service in the Internet has led to various works which attempt to work around a partial deployment by using the existing unicast infrastructure where appropriate.

4.2 Identifier/Locator Split

The distinction between endpoint identifiers and network locators was discussed in §2.4, but the current Internet architecture confuses the two (§3.4.1). The lack of isolation between layers in the network stack ties identifiers used at the transport layer to locators used at the network layer, requiring that transport layer identifiers are modified when an endpoint changes its location in the network. Also, exacerbation of BGP forwarding table sizes, primarily due to multihoming and the adoption of PI addresses as company-owned identifiers, violates the hierarchically assigned locator semantic of the network layer address. Arguably, the tight coupling of layers by an IP address has also hindered the adoption of IPv6 due to the required modification of both transport and network layer protocols to accommodate the new network layer protocol.

Some alternative network designs have chosen to focus on the identifier/locator split to decouple the network and transport layers in the protocol stack (potentially aiding host mobility), to provide novel solutions to the BGP forwarding table scalability problem, and to possibly also ease a transition to a new network layer.

The work on the identifier/locator split can itself be split into two categories: host-based solutions, and network-based solutions.

4.2.1 Host-based

Host based solutions aim to provide a identifier/locator split in the hosts, without affecting the network infrastructure. There are various host-based solutions which appear viable.

The Host Identity Protocol, HIP, introduces a new namespace and a new protocol layer between the network and the transport layers; HIP does not propose to replace the existing IP system [80]. HIP introduces a new namespace for the transport layer, and provides a clearer distinction between the layers of the network stack.

A host identity may be the public part of a public-private key pair; using the host-identity as the input to a hash function to generate fixed-size identities of either a 128-bit “Host Identity Tag” (HIT) or 32-bit “Local Scope Identifier” (LSI, to allow the possibility of migration for older applications using 32-bit wide addresses) provides fixed-size fields for use in HIP packets.

With the transport layer using host identities, hosts should be able to roam freely without terminating existing data flows, provided there exists a good enough mapping service between identities and IP addresses, or some other rendezvous service. For HIP identities less likely to move frequently, a new DNS entry type to map identities to IP addresses may be viable.

Given that HIP does not require network modification, its deployment could be gradual, in a host-by-host manner over time. HIP, if widely deployed, may allow RIRs to discourage the use of PI address blocks, since one of the primary reasons for these address blocks has been to use them as a form of identify. Without affecting the network layer, it does not diminish the desire for ASes to multihome; ideally, a host in a multihomed network would receive one hierarchically derived IP address from each parent network, and the HIP layer could decide which one to use.

HIP could operate over IPv4 or IPv6, and may even allow end hosts using the different network layers to communicate with each other. HIP, however, makes no provision for multicast.

Shim6 is another approach to offering multihoming without the use of PI addressing [81]. When using Shim6, end hosts within a site are aware of the various IP addresses they have been assigned, corresponding to the prefixes assigned by the parent sites. When initiating a communication, the host chooses one of its network locators, after which the "shim" layer allows the network stack to present that locator to the transport layer as an identifier; the network layer can switch between the available prefixes to reflect service availability or policy change, without having to reset transport-layer associations. Shim6 only operates with IPv6.

4.2.2 Network-based

Network-based identifier/locator split solutions require some network modification, but aim to avoid modifying the end-hosts themselves. Like host-based solutions, there are various network-based solutions which have received attention.

GSE proposes an alternative way to structure IPv6 addresses, in three parts: the end system designator (ESD); the site topology partition (STP); and the "routing goop" (RG) [82].

The ESD is the "identifier" part, uniquely identifying a host; the RG is the locator part, drawn from the hierarchically assigned portion of the address space in the public Internet; and the STP part is used to describe a subnet within the AS that the RG resolves to.

The End System Designator (ESD) is an 8 byte, fixed size, globally unique identifier for the node to which it belongs. The ESD represents an end point (though each end system may have multiple ESDs); a suitable ESD may be a MAC address, but dynamically allocated ESDs for links which do not have a suitable MAC address to use are also required.

The RG portion of an address is re-written on packets entering and leaving a network: on a packet leaving the origin network, the RG locator of the network is written into the source address field of the packet header; on entering the destination network the RG locator part of the destination address is set to a "site-local" value, not routable across

the public Internet, after which the STP and the ESD are used to route the packet to the endpoint.

LISP, the Locator/ID Separation Protocol, suggests a "map and encap" (map and encapsulate) scheme for IPv4, where end hosts are identified by endpoint identifiers (EIDs) and routing locators (RLOCs) are distributed hierarchically from AS to AS [83]. On routing a packet from one EID to another, if the EID does not exist within the same AS then a lookup takes place to discover the current RLOC of the AS in which it resides ("map"), and packets are then essentially tunnelled ("encapped") across the inter-network routing infrastructure.

By enforcing hierarchical RLOC assignment, LISP would be able to solve the multi-homing issue; the mapping function can simply return two or more valid RLOCs for the EID looked up, the choice of which to use being on the originator network.

LISP requires no changes to existing hosts, and little additional hardware in end-sites to facilitate the mapping from an EID to a current routing locator RLOC, and subsequent encapsulation of packets sent to that RLOC. LISP also doesn't require modification of the IPv4 core of the network.

The unsolved part is the mapping function. Different LISP variants are proposed which differ only in their implementation of the mapping service from an EID to a current RLOC. LISP doesn't consider multicast, aside from suggesting that LISP continue to use the same class-D address space, but as both EID and RLOC. It proposes no solution for group location, or forwarding mechanisms, etc. Massey et al proposed a very similar scheme to LISP, which uses similar ideas albeit under different terminology [84].

Each of the schemes presented here, both host-based and network-based, seem to offer a reasonable way to provide a locator/identifier split in the current Internet architecture. The common unanswered question is the mapping function between those identifiers and locators, though it is quite reasonable to expect that DNS could be used for many nodes, perhaps alongside a similar mechanism dedicated to more mobile nodes.

4.3 Alternative Routing Architectures

The Internet Indirection Infrastructure (i3) proposes a rendezvous-based communication abstraction, where a source sends packets to an ID, and receivers subscribe to that ID to receive packets; this indirection makes implementation of multicast or anycast reasonably easy [85, 86]. IDs are placed in a Chord-like overlay, and packets bound for an ID are forwarded to the Chord node responsible for that ID, which forwards packets to their destination(s).

FARA discusses the general concepts behind entities which may wish to communicate on: entities, associations, communication substrate, rendezvous, the FARA directory service, and "slots" [87]. FARA acknowledges that they have attempted to extend it naturally to include multipoint communication, albeit unsatisfactorily.

NIRA (the New Internet Routing Architecture) aims to encourage competition in the ISP market by allowing users choice in the routes their packets take, and which networks a packet will traverse [88]. Protocols would be in place for a node to discover

its "up-graph" toward the core of the network, and also to query the destination's "up-graph", allowing choice in which networks a packet will traverse while in transit. The chosen route is then encoded into packet headers. NIRA does not discuss multicast routing or addressing.

HLP aims to ease the volume of routing updates observed in the current BGP system by exploiting the hierarchy of provider-customer relationships across most of the Internet graph and employing explicit information hiding, such that minor updates within one hierarchy will not be visible within other hierarchies [89]. BGP, as a path vector protocol, exposes path information to the rest of the Internet. HLP utilises a link-state protocol within a hierarchy to reduce levels of churn and improve convergence times, and a distance-vector protocol to communicate between hierarchies.

4.3.1 Overlay Multicast

There has been much research on exploring new routing methodologies, over using the existing IP network; such network overlays have oft been used to implement multicast in the absence of network support. These systems are described in this section.

Overlay networks constructed on top of the standard IP protocol have been able to provide either multicast functionality, or alternative naming schemes, or both.

Ring-Based DHTs

Pastry addresses peers with a 128-bit node ID, which occupies a point in a 1-dimensional circular node ID space [90]. Pastry nodes can route from one to any other in $O(logN)$ overlay hops by keeping pointers into the node ID space at certain points (e.g.: at one half, one quarter, one eighth of the way around the namespace). Routing tables are of size $O(logN)$.

A multicast system called Scribe has been built to operate over a Pastry substrate [91], and another called SplitStream for high bandwidth content distribution [92]. Scribe uses a rendezvous node and explicit *join* messages, much like (O)CBT, to set up a shared multicast tree for a group. SplitStream splits content into "stripes", each stripe having a separate distribution tree. Distribution trees are organised such that each node is only an interior node in one, and a leaf node in all others, such that the workload is distributed evenly across all overlay peers, and all nodes contribute some upload bandwidth to the group. SplitStream uses Scribe to build its distribution trees.

Chord is a system similar to Pastry, in the sense that nodes are mapped onto a 1-dimensional coordinate space, [93]. Chord, like Pastry, achieves routing across the overlay network in $O(logN)$ hops.

Ring-based topologies are easy to understand and easy to implement at the application layer in the current Internet. They do, however, tend to ignore topology, minimising their potential as routing protocols at the network layer.

Plaxton Meshes

A Content Addressable Network (CAN) operates over a d-dimensional Cartesian coordinate space constructed on a d-torus [94]. CAN is based on work by Greg Plaxton

[95]. Objects are placed in the CAN space deterministically according to a uniform hash function over some key, K. Nodes in a CAN are allocated sections of the space, with any keys falling into their space belonging to them. Routing tables are small ($O(d)$ in a d-dimensional CAN space) because nodes only need know about direct neighbours; routing involves attempting to take the shortest path toward the destination (in essence attempting to draw a straight line from source to destination). The average path length between any two nodes in a CAN is $O(d(N^{1/d}))$, where N is the number of nodes in the system.

Any-source multicast over a CAN is achieved by flooding data across the CAN; mini-CANs can be created over a larger CAN space to generate smaller groups [96]. The coordinate space of the CAN allows for efficient flooding algorithms to be implemented which reduces the number of duplicated packets to (almost) nil.

Tapestry is another structured peer-to-peer overlay based on Plaxton's work. Routing in a Tapestry system is a matter of matching as much of the destination ID to an neighbouring ID from a given node, similar to longest prefix matching used by IP routers [97]. Neighbour maps are, however, of constant size, and each node in the path between source and destination only has to match one further digit of the Tapestry ID. The distance a packet travels in Tapestry is proportional to the the distance travelled in the underlying network, suggesting that Tapestry is at least reasonably efficient.

Bayeux provides single-source multicast over a distribution tree built using the Tapestry substrate for routing purposes [98]. Signalling messages are sent toward a source to set up the forwarding state as required. The simplest way to achieve multi-source multicast using Bayeux would be to build numerous distribution trees, each one rooted at a different multicast source.

Other Overlays

Kademlia is used as a lookup substrate in real-life file sharing applications; it organises nodes into "k-buckets" (where 'k' is the size of each bucket, chosen for the application), with each bucket containing different portions of the address space. A Kademlia node holds multiple k-buckets, with each k-bucket holding k entries to different nodes; each k-bucket contains nodes at different distances according to the XOR distance metric [99].

CollectCast [100] is a system built on top of Pastry designed for streaming content. It selects peers from a candidate set of nodes holding the content to be distributed, based on available bandwidth and packet loss rate. The active sender set is the subset of those in the candidate set who can provide the highest quality playback for the receiver. Collectcast utilises the lookup substrate beneath it to fetch locations of required content; the implementation of Pastry used by CollectCast was modified to return multiple references to a given object, rather than just one. PROMISE is the name of the actual implementation of the CollectCast system, and demonstrates high levels of performance, and reliability [101].

Again, each of these overlays is capable of implementing some multicast functionality, albeit at the application layer, with the overhead costs required to maintain such an overlay.

4.3.2 DHT-Inspired Internet Architectures

'Routing on Flat Labels', ROFL, draws on DHT work to present an Internet architecture which does not require a hierarchical addressing structure [102]. The two projects ROFL draws on are Virtual Ring Routing (VRR) for intra-domain routing [103], and Canon for inter-domain routing [104].

VRR is a ring-based DHT system, intended for ad-hoc wireless networks, but employed by ROFL as on intra-domain protocol. In ROFL, routing takes place over identities alone, abandoning any sense of hierarchical network locators. The routers maintain successor and predecessor routing entries for all the nodes that it "owns" within the AS (i.e., end hosts which consider that router to be its default router), which creates (at the very minimum) a ring-structure; routers will cache other routes that hosts are using. Forwarding is greedy, in that a node will route toward the nearest ID, but not beyond, when forwarding a packet.

Canon allows construction of hierarchical DHTs, thus allowing for a hierarchical graph of autonomous systems somewhat similar to the current Internet graph. The Canon routing protocol guarantees that a packet will not leave an AS if the destination is contained within the same AS.

ROFL as an Internet architecture is suitably different from other Internet architectures in the sense that it does not attempt to use network locators (in the hierarchical sense, at least), and does not attempt to always construct shortest paths between endpoints. ROFL is interesting in this sense, but lacks knowledge of actual topological information. Node joins seem to be expensive also, and while forwarding tables are small, packets may traverse many links to reach their destination.

4.4 Compact Routing

Intuitively, shortest path routing can move a packet through – and out of – the network as quickly as possible; so if we reduce the number of packets in transit, then we alleviate the bandwidth requirements on the links in the network. However, shortest path routing leads to increasing routing table sizes as evidenced by the rapidly expanding BGP forwarding table sizes, and has been highlighted as a flaw in the current incarnations of IPv4 and IPv6 for some time.

Compact routing is a loosely defined research area looking at routing algorithms which aim not to provide shortest path routing, but routing characteristics approximating shortest path routing; the actual path taken by packets will probably not be the shortest path, but a compact routing algorithm dictates the worst-case "stretch" of a graph as being the largest multiple of hops from the shortest potential path that a packet will take to get to its destination [63, 105, 106]. By deviating from the standard goal of shortest path routing, compact routing schemes are able to reduce forwarding table sizes significantly. This is important: as forwarding tables increase in size, so too does the time to analyse the table *per packet*. Memory access speeds are not increasing at the same rate as the routing table and at some point a router with a full BGP table may not be able to forward packets at line-rate, leading to filled buffers and dropped packets.

The compact routing work asserts that it is not possible to develop a truly scalable and efficient routing algorith unless the requirement that packets are routed along shortest

paths is weakened. This work looks like a feasible way of reducing state requirements in the BGP system today. Interestingly, they assert that on simulated graphs similar in shape to the AS graph, average the stretch on a "stretch 3" algorithm was 1.1, and so while the worst-case stretch with such an algorithm is 3 multiples of the shortest path, the average case is extremely close to the shortest path. As yet, however, there has been a lack of experimental evaluation of these protocols.

4.5 Heterogeneity of Networks

Plutarch argues that the common network layer which has made the Internet successful is not necessarily useful in the modern environment, with many low powered or mobile devices now available [107]. Plutarch suggests that future network architectures should embrace this heterogeneity of networks.

The Plutarch architecture calls two things: for contexts, which are homogeneous networks in some regard, with a consistent naming and addressing scheme; and interstitial functions, which modify data passing between two contexts. One existing example of this is NATs, which translate between contexts of either side. The work seems vague in terms of how names are resolved, and in particular how a target context is located, given that there are no global names. The lack of a common network layer, and the insertion of the hardware required to communicate between contexts would also violate the end-to-end argument, though this may not be impossible to overcome (see §4.6.1).

Turfnet is a similar scheme, but one which requires a global addressing scheme (and global name-resolution service) to achieve "inter-turf" routing [108]; a "turfnet" is an autonomous network, which can dynamically connect to other turfs.

The Unmanaged Internet Protocol (UIP) makes the (reasonable) assumption that the Internet will evolve into a network with a largely static infrastructure portion, with high levels of mobility around the edges [109]. The UIP, then, aims to support *identity-based routing*, i.e., routing on flat identifiers. In a similar manner to HIP, UIP inserts a layer above the IP layer which uses hashed cryptographic keys to provide flat names and some additional security, but UIP routes directly on the flat labels via a DHT-inspired routing mechanism. The aim is to achieve not shortest-path routes between nodes, but routes with a reasonable worst-case stretch factor, and reasonably sized forwarding tables also. UIP, however, provides no multicast provision; the work is only discussed in a position paper, and so is scant on details.

4.6 Miscellaneous Architectures

The virtually invisible Internet suggests a group-based architecture which, rather than offer a multicast-enabled Internet, aims to generalise the network structure such that the fundamental organisational primitive is groups [110]. In this sense, mobility would be supported by the network directly, and routing between nodes within a group would still take place even while mobile. (In a sense, it sounds like lightweight, automatic VPNs.) The work is light on ideas for implementation, unfortunately.

4.6.1 Middlebox Architectures

Some work aims to assimilate the existence of NATs and firewalls into the network architecture, deeming them important parts of the infrastructure despite breaking the purist end-to-end model.

The IPNL (IP Next Layer) architecture aims to isolate different addressing realms by providing mechanisms [111], but has been superseded by similar ideas in a project called NUTSS.

NUTSS aims to embrace middleboxes by employing both an off-path system and an on-path system for connection establishment: the off-path system is used between networks to negotiate connection characteristics, and if successful in negotiating to exchange keys used by the on-path signalling protocol between middleboxes to explicitly open a connection between, say, two firewalls [112, 113].

DOA, the Delegation Oriented Architecture, facilitates deployment of middleboxes by allowing end-hosts to define which intermediaries a packet should traverse [114]. This, for example, might allow a host to use one firewall service in particular for all packets coming from a certain network. The DOA layer sits between the network layer and the transport layer, and allows hosts to have a globally unique name in a flat addressing space; the DOA therefore requires a mapping service to map endpoint ID to an IP address.

4.6.2 Higher Level Designs

There exist various other ideas which diverge from the main topics covered already.

Active naming suggests that end hosts be aware of two parts of an address: the address itself, and a standard name of the resolver which knows how to parse that address when deciding how to forward the packet [115]. (For example, a URL and a program which understands how to parse URLs.)

The Role-based Arch formalises the various stack requirements for an application, and generalises the network stack away from the rigid TCP/IP stack in the current Internet, allowing for greater levels of flexibility in protocol design [13].

TRIAD offers a content-based naming and routing scheme, where packets are routed by names rather than IP addresses. Uses name identifiers for node identification and routing [116]. The Data-Oriented Network Architecture (DONA) is another scheme superficially similar to TRIAD, but allows for new primitives over the Internet, such as a *find* operation to locate date, or a *register* operation to retrieve data, much like a publish/subscribe system [117].

4.7 Discussion

Many of the alternative network architectures presented here aim to alter the unicast model in some way, either with different unicast routing protocols, or by offering a clear identifier/locator split. These often choose to focus on unicast behaviour, though, and do not consider multicast behaviour.

However, the alternative multicast protocols presented in §4.1 offer methods of providing multicast in the network without the requirement of on-path state at all intermediate nodes between sources and recipients. Indeed, Xcast requires no additional state in the network, merely extended packet headers and routers that understand them (though this is not a general, scalable solution). These, coupled with the various overlay networks also discussed, suggests a strong interest in offering group communication across the Internet.

Overlay networks have also allowed exploration of new network architectures, and alternative methods of utilising endpoint locators. The work relating to the identifier/locator split demonstrates the utility of new naming and addressing schemes in the network stack.

The overlay networks and the compact routing work discussed also shows a willingness to explore non-shortest path routing.

The works covered in this chapter demonstrate the various directions network research is taking. The various ideas here provide some interesting ideas worth exploring, which will be covered in Chapter 5.

Chapter 5

A Group-Centric Internet Architecture

Given the common network use-cases described in §1.1, each of which uses some form of group functionality, and the various network overlays designed to provide various forms of group communication in §4.3.1, there appears to be a lot of duplicated effort invested in providing group functionality to a broad range of applications, suggesting that this is common functionality which could be better placed in the network.

This maps onto the sheer volume of research into IP Multicast, suggesting that allowing the network to replicate packets sent from one host to a group of recipients is a desirable goal. It adheres to the end-to-end principle of placing functionality in the network when it is beneficial to do so. There have been many concerns stated with regards to the suitability of the Deering multicast model to the wider Internet and, as such, and coupled with various shortcomings in the current Internet architecture, I feel there is scope to explore an architecture which places group communication as its primary focus.

In this chapter, I'll present initial ideas for a design of a new network architecture; I'll cover some basic assumptions and requirements for a new architecture in §5.1. I'll use these requirements in the subsequent sections: §5.2 covers routing and addressing, and §5.3 covers packet forwarding to group members. I'll offer some brief discussion of stack layers §5.4, before a summary of how the architecture currently looks in §5.5.

5.1 Assumptions and Requirements

For the purposes of considering an alternative Internet architecture, some initial assumptions must be presented:

1. From the discussion in §2.1, I shall assume the best approach for offering a new Internet architecture is to build it as a datagram forwarding service. Virtual circuits are another possibility, but the requirement of on-path signalling state, and not knowing how many groups might exist in the network at any one time, suggest that datagram forwarding makes more sense.

2. There are multiple independent organisations cooperating through the use of common network protocols to offer the Internet service. Inter-domain routing takes place across a network of autonomous systems. We must assume heterogeneity in terms deployed hosts, routing hardware, link-layer network mechanisms, and available bandwidths.

3. Unicast can be implemented as a special case of multicast, and therefore viewed as the same service from the perspective of the applications. Unicast, then, is a two-party multicast group where both parties are permitted to send data into the group.

4. A new architecture may require new hardware to be deployed within ASes, in end hosts, or new software be deployed, though it is desirable if the new functionality can be tunnelled through parts of the network not yet enabled.

Requirements for a group-centric Internet architecture based on these assumptions, the shortcomings of the Deering multicast model and current Internet architecture are as follows:

I. Clear decoupling between transport & application layers (code which resides at endpoints), and the network layer (the common protocols between endpoints), by providing endpoint identifiers and network locators respectively. The decoupling allows for greater flexibility at the network layer and should help it to evolve separately from the transport layer. An application should not need to know its network locator to be able to operate, though it may be useful to know when its location has changed such that transport layer associations can be reset.

II. Implied in point 3 above, is that the network should allow for transparent variability in group sizes; a group containing two members should be able to admit a third member with ease, if it makes sense to do so (see assumption V(b)).

III. No on-path state (or, at best, minimal on-path state): The network should be capable of reacting to changes in network conditions and topology, and able to route around problems.

   (a) One of the key problems with on-path state is that all intermediate nodes between sources and destinations must be functioning to offer service to all recipients. On-path forwarding state is rigid, and requires redundant state between the source and recipients. It may be acceptable to offer a service where forwarding state retained in the networks grows relative to the size of the group.

   (b) Unicast protocols are capable of adapting to network conditions. Dense-mode multicast protocols are also able to adapt to network conditions, by repeated flooding or total knowledge of the network. Sparse-mode protocols are less able to react to network conditions given that they rely on on-path state instead.

IV. A lightweight join/leave mechanism, and lightweight group creation/destruction. Any Internet architecture aiming to remove the distinction between unicast and multicast semantics must be able to support many small, potentially transient groups.

V. Security: One of the perceived flaws with the Deering multicast model is that any node can send packets into a group without being a member of that group; likewise, any node can join a group. This suggests:

   (a) It may be beneficial to require a node to connect to a group prior to any data being forwarded by the network to that node.

   (b) Related to this, it may be beneficial for a group owner, or group initiator to be able to specify the size or behaviour of a group, e.g., 2 party group; any number of receivers but one source; authorised source list, etc. Should the source have fine-grained control over who can send or receive packets.

   (c) To aid the network in building distribution trees, a connecting node may also specify that it wishes to only send packets into a group, that it wishes to send and receive, or that it wishes only to receive.

   (d) If it is to be possible to deny a node entry to a group, then the behaviour of the "connect" operation must be specified either in the network layer, at a rendezvous node, or at the group source. The difficulty in implementing such a scheme lies in determining the "owner" of a group.

   (e) There may be a case for allowing group membership to be group visible to group members, if appropriate. Some groups may require that only a subset of members, or perhaps only the group owner, can view membership. Other groups may require anonymity for its members.

VI. Congestion control solutions for multicast groups involve layering mechanisms, such that data is striped and sent across multiple layers. Endpoints subscribe to as many layers as possible before packet loss becomes apparent [118, 119]. As a method useful for congestion control, it would make sense for layering to be an integral part of a group-centric architecture.

The requirements presented here are all useful requirements for an Internet service which emphasises group communication. The following sections consider key areas of a new design in light of these assumptions and requirements, and build up from the network-layer concerns to some of the higher layers in a traditional protocol stack.

5.2 Network Addressing and Routing

Given that I am designing primarily to support shared trees, the routing and addressing design of such an architecture may benefit from an addressing scheme which helps forward control packets relating to a group toward the source or owner of that group, or rendezvous point of a shared tree. It may also be used to forwarding data toward recipients, depending on how the forwarding mechanism functions; the forwarding schemes are influenced by the chosen addressing scheme and *vice-versa*, but I shall eliminate some addressing possibilities and cover forwarding in §5.3.

Also, since this design is for an inter-network, addressing and routing mechanisms suitable for an inter-domain environment must be considered, and how these affect the routing information required to generate forwarding state at nodes.

As discussed, an endpoint must join a group prior to sending any packets into, or receiving any packets from, that group. This implies that either:

- The forwarding mechanism will not have enough knowledge to forward packets if a join has not taken place, implying signalling state.
- The forwarding mechanism will have enough information to know to *not* forward packets, implying that group membership state is pervasive.
- The forwarding mechanism will forward packets toward a group until some intermediate point whereby the packet gets dropped (this intermediate point might be an edge router on the AS containing the end host)

The flat IPv4 Multicast address space is at odds with the hierarchically assigned unicast address space. There is no correlation between a group and its receiver-set, or even its set of sources; the flat addressing scheme offers no real sense of address ownership. The reasonable goal of providing some form of security or access control to the group, however, suggests that there must be an entity in the network capable of handling these control-level requests for each group further implying a clearer sense of ownership is required to simplify a group-centric design.

If there are conceptually two modes of operation, that of sending data (or *join* requests) into a group, and that of receiving data back out of a group, then a topological mapping for the latter is difficult to achieve (given multiple occurrences of nodes wishing to receive data from the same group). The former, however, can make better use of a locator in the unicast sense. We may also ease the concept of group ownership by designing the network layer such that control requests are naturally forwarded toward the "source" or group owner by using a locator "owned" or managed by that node.

Possible options to achieve this, then, are:

- Flat addressing, where the group address (e.g., a 32 or 128 bit uniquely identifying a group) is advertised out into a BGP-like system. This option carries the obvious burden of holding a lot of state in the routing system to compensate for a lack of structure in addressing. This option does not directly solve the address allocation problem inherent in the current multicast address space.
- GLOP-like allocation, where each network is allocated a large portion of address space from which group addresses can be assigned to endpoints. This requires coordination between an end-host and the parent network to allocate a group locator for each new group, but carries the advantage that the host ("owner") address is not exposed; it creates an indirection point between a group source and receiver set.
- GLOP-like allocation, but where a host has its own dedicated address space from which it can assign group addresses. The locator for the group is actually the GLOP part plus the host part, with space free in the address structure for the host to use as group numbers. Removes the requirement to coordinate with the parent network for group address allocation, but exposes the source node's location in the network. A rendezvous mechanism may still be able to intercept packets bound for this address in a shared tree, or for control traffic.

An SSM model of channel addressing may be possible on either of the first two options; the third option is similar to the SSM model already, but without the distinct *source* and *channel* parts.

The compact routing work suggests that routing on flat labels is not infeasible, and therefore the first option might also not be infeasible. However, there is a lack of experimental work in real-life networks on these protocols, and it may be that the propagation delay in advertising a new group across the entire network render this unusable as a mechanism for setting up fast, lightweight groups. Further investigation would be required.

The GLOP-like mechanisms in the latter two options suggest a stronger sense of group "ownership", which maps well onto many of the group applications which we might want to run over the network. It may also make some of the security and group admission requirements easier, aiding requirements IV and V. An addressing mechanism such as this removes the need for an inter-domain group discovery/group address management system.

The second option would still require an intra-domain address management system; the third option, on the other hand, implies that each node is given a quota of addresses, from which it can generate groups.

In these latter two options, a hierarchical, provider-assigned network prefix might be used rather than GLOP addressing, but the end-result is similar: There is a network portion of a locator, and a group or host part to the address; the main point is that there should be a mechanism for routing toward a group's "home" network using the PLOP or network-prefix part.

Of these addressing schemes, the last seems the simplest to manage.

There are few group applications where the owner of the group would wish to depart from the group long before the group itself has expired, and these may cause problems for ASes who do not wish to manage groups when no members exist within that AS itself. These applications may be managed in an application-by-application basis, where each group member initiates its own group (and therefore its own distribution tree), and an "any source" group can be constructed from multiple "source specific" groups. In this sense, multiple groups may be managed from the transport layer to provide an any-source service, without group ownership issues lingering at ASes who no longer have anything to do with the group. This may ease some concerns regarding accountability.

5.2.1 Supporting Group Layers

Requirement VI states that a new architecture requires some form of congestion control, and therefore an awareness of the layering mechanisms used to control congestion within groups. Layered groups are useful for offering congestion control in group settings. Also, there may be situations where it may make sense to declare subgroups within a group, which a layering mechanism may be useful for. (e.g., owners or administrators of a media stream may wish to receive not only the streamed data, but also some application-level management feedback from the group which the viewers themselves should not be privileged to see). Such a layering mechanism is natural concept for groups, in that the various layers are all related to the same group communication.

The actual implementation of group layers, however, may be best left as a transport layer concern, or adopted as a network layer concern. There may be a case for providing some network layer support at the network layer, to support the endpoints. As such, possible methods for handling group layers are as follows:

1. There may be no network support, with end hosts managing the layers manually via separate groups. A lack of native support for group layers in the current multicast model required that work investigating these congestion control mechanisms has used separate groups monitored at the endpoints. There isn't a natural relation between layers when using this mechanism.

2. Another option is to use different port numbers for layers within the same group. This would require that the network be aware of ports, if the end hosts were to be able to subscribe and unsubscribe from those layers on a port-by-port basis.

3. Similarly, the SSM model of channels could be adopted, such that layers are distributed over channels.

4. Group layers could be embedded directly in the address structure, such that if 16 bits of "layer" space is set aside, then the first N-16 bits of the packet can be used for easy routing toward a source; the full N bits can be used by routers to forward packets to receivers.

The primary advantage to offering group layer support in the network is that there would not need to be any additional per-layer state retained, which may be more heavy-weight than a congestion control mechanism requires.

The latter three options imply that a "connect" operation may only have to occur once, and individual layers are subsequently "enabled" or "disabled". This may require an additional switch inside routers holding forwarding state, such that very little additional state is required to enable or disable individual layers within a group.

If a layered approach is to be utilised also as a useful "subgrouping" mechanism, then we may require that the network, rendezvous mechanism, or source be able to dictate who is capable of joining a layer. This may be useful functionality, but will not be covered in this report.

5.2.2 Summary

Two primary concerns have been covered in this section, in terms of addressing: how to use addressing to ease address ownership issues, and how to use addressing to support layers, or subgroups, to enable congestion control.

The most obviously useful and workable solutions available currently appear to be the GLOP-like system, where end hosts are awarded a locator by their parent network, allowing end-hosts to allocate group addresses from this. With group layers also embedded in the address structure, a network locator in this architecture will include:

- AS number (32 bits).
- Host number (e.g., 96 bits).
- Group number (e.g., 32 bits).
- Group layer number (e.g., 32 bits).

This address structure then totals 192 bits, which would appear to be large enough to support most future applications. 2^{96} end host identifiers is a wide enough address space that these could be assigned uniquely, or they might be assigned by the parent AS. 32 bits of address space for groups is a large enough space that end hosts should not be afraid of creating even transient groups, and a further 32 bits for group layering surely covers many future application types and layering schemes.

The GLOP-like addressing scheme suggests that if a rendezvous node is required, then it'll also be hosted within the same AS. That is, if a rendezvous is responsible for group admission, etc, then in may be desirable to have it intercept certain types of traffic, decoupling a source from the rest of the group.

5.3 Network Forwarding State

A fundamental aspect of a group-centric architecture is how it delivers packets sent into the group to all group members. A group-centric architecture requires a mechanism which will duplicate packets where necessary, but will minimise on-path state, preferably utilising no on-path state at all, as per requirement III.

The forwarding mechanism may be required to handle both shared distribution trees, possibly bidirectional, and source-rooted, shortest-path trees, depending on the application. Certainly the most important of these is the shared tree, in the sense that most applications do not need to meet real-time responsiveness constraints, and a shared tree is perfectly acceptable for many applications. Many groups will be able to utilise a shared tree without a perceived loss in performance.

There exist a spectrum of possibilities for state placement to facilitate inter-domain packet forwarding to group members. In decreasing order of additional state required in the network:

1. Pervasive state: All ASes know of all active groups, and all receivers within those groups. With AS-level forwarding state such as this, the network should be able to forward packets to all receivers without additional signalling state; inter-domain routers simply forward packets to all ASes advertising the group address. The paths that packets take should be able to react to changes in network conditions. It would allow the routing infrastructure to route around problems and route across policy setup without having to be re-initiate a connection.

   This may be restrictive however; if fast joins are required to support many transient groups, then state for a join must be broadcast across the AS graph. This is a solution which requires no fixed on-path signalling, but it is certainly not a lightweight solution.

2. On-path signalling. PIM-SM is widely used within the current intra-domain infrastructure, and requires on-path state, as does PIM-SSM; these are both complex protocols, in part because a node may choose to switch from a shared tree to a source-rooted shortest-path tree during the lifetime of a connection. (O)CBT also uses on-path state, and it offers only shared trees, though it has not seen widespread deployment. As per requirement III, on-path state is undesirable.

3. On-path state only at divergent points, however, would reduce the burden of redundant state between sources and receivers, and would at least allow for variability in packet forwarding between those divergent points.

4. Multiple destinations listed in packet headers. Multiple destination packet headers would be manageable only on small groups; the forwarding infrastructure may be intelligent enough to compare the destination list against a forwarding table, and understand when to duplicate packets and when to not. It requires that endpoints maintain lists of group members.

Of these, the first option seems too heavyweight a solution to consider seriously. The second solution requires too much on-path state, much of which may often lie redundant, and more resembles virtual-circuit switching.

The third option was covered briefly in §4.1, with discussion of the REUNITE and HBH concepts. These protocols do require additional *control* state be kept, but *forwarding* state grows with respect to the size of the group. Forwarding state is placed in the network only where packet diverge. This seems to be a useful option to consider.

The fourth option was also covered in §4.1, with Xcast. This option obviously doesn't require additional on-path state, but can only be feasible for small groups.

5.3.1 Forwarding and Group Layers

There may be scope for a prioritised best-effort system, where some data streams can be awarded a lower priority than the best-effort "standard" priority; this may be useful for the congestion control mechanisms already mentioned.

The network might be able to use these group layers to prioritise queues at routers, with lower priority layers more likely to see packets dropped if queues in a router are nearing full, implicitly signalling to the end host that the network may be unable to accommodate some layers, and so to back off.

5.3.2 Summary

The most realistic forwarding mechanism out of those described here seems to be the HBH scheme introduced earlier. A scheme like this has several key benefits:

- Required state is set up explicitely at the initiation of the group, and does not require the heavyweight schemes mentioned initially.
- Forwarding state grows relative to the group size.
- The "core" of the inter-domain network need not contain any HBH routers – it can remain a simple unicast service, and therefore needn't be hindered by rewriting or duplicating packets. The network core therefore need not carry forwarding state to implement multicast, and so the "core" of the network remains a simple point-to-point packet forwarding system.

The use of an HBH-like scheme suggests that the "dumb network" ideal that packets will be routed entirely independently of each other according to network conditions is half-broken here; changes in network conditions are reflected in the paths taken by packets in transit between HBH routers.

From an inter-domain context, I expect that HBH routers would be a key component of an AS, and that the HBH-like algorithms would operate over the AS-graph when duplicating packets, creating a distribution tree from that AS-graph. In this sense, packets may still route around problems between ASes. Also, ASes between divergent points being used for transit need not hold any additional per-group state.

Control plane state per node in this sort of system may be high, but forwarding state grows with the size of the group. This may be acceptable, in that control state need not be accessed at line speed, and so routers may be able to hold control state in large, but cheap, memory stores. Forwarding state for a two-node group in this sort of system is no greater than unicast forwarding tables: the source sends packets to the only destination address. This also maps well onto requirement II, if we allow unicast communication to occur over these groups, then we have a system where a third group member can be admitted with ease.

An ideal system could be imagined where path signalling is used as a method of *optimising* the paths that packets take from a source to a destination, provided that the signalling is not *required* to maintain a packet flow.

Network layer awareness of group layering, allowing packet drop mechanisms to be implemented, will provide additional implicit feedback at the endpoints, aiding their decisions regarding which layers to subscribe to and which to not.

5.4 Application and Transport Group Definitions and Semantics

Given requirement I, regarding the identifier/locator split, a group-centric architecture requires an ID at endpoint which is distinct from the locator(s) the network uses to locate group members; at the transport layer, a group identity should relate to the existence of a group, and not to the group's current membership or location in the network. The network locator may embed additional information to aid packet forwarding, but this should not necessarily influence the transport-layer identifier for the group.

Useful elements to consider at this point include basic API calls, and what sort of data these calls may require.

5.4.1 Transport Layer

The transport layer must offer to the application developer a connect call, which requires as parameters:

- A group identifier. This may be the 128-bit combination of the end-host number and the group number used at the network layer, or it might be an entirely identifier, entirely unrelated to the network locator.

- The expected usage the application expects to make of the group: it must also be able to specify read/read-write/write behaviour, analogous to file-system calls. It may be useful to assume "read-only" privileges if this parameter is omitted.
- Required connection characteristics, such as reliable data transfer, or unreliable data transfer. This influences the transport layer for a group, and should not influence the network layer at all.

The connect call may return a socket descriptor, as used widely in the current sockets libraries. With no equivalent of a *sendto* call allowing an application to send data into an arbitrary group, the transport layer then need only offer a send and a receive call to the application. These then take as parameters the socket descriptor, the data to be sent (or space within which to receive data), and the layer to send data on.

In supporting sublayers, then the application won't necessarily be aware of which group layers a packet arrived on. Some applications may benefit from having this information though, and so it should be available if the application wants it.

A disconnect call must be present also, to enable the network to remove any state which had been generated for that recipient.

5.4.2 Transport Layer Protocols

Transport layer protocols for multicast groups have been researched, but have never reached the same level of ubiquity or maturity as the TCP/IP and UDP/IP split seen in IPv4 unicast. There are, however, protocols which map well onto group scenarios, including: RTP (RFC 3350); protocols for delay invariance [4, 5]; PGM, Pragmatic General Multicast, (RFC 3208).

5.4.3 Network Layer

In terms of broad API calls a transport layer might require from the network layer, I expect simple calls conceptually similar to the following:

- A connect() call, required before packets can be sent into, or received from, a group. So the connect call probably requires: a network locator; a read/write request similar to filesystem usage, indicating whether the node wants to join as a listener, a participant, or both. This call will probably return something like a socket descriptor, to be used in subsequent calls.
- An equivalent close() call, which accepts the socket descriptor as an argument, and terminates contact with the group; it allows the network to destroy whatever state may have been set up for the communication, and also allows for group membership of the node to be revoked at, e.g., a rendezvous point or source.
- A send() call, which accepts a packet and sends in into the group; the send call will fail if write access to a group was not requested, or write access was not permitted.
- A recv() call, which listen for an incoming packet, to pass up to the transport layer.

The connect call will create an association between a socket descriptor and a group address; it will also prompt whatever network-layer protocols are required to set up required state for the group. The close call will teardown any state.

When receiving data, the transport layer (*not* the application) must be able to modify which group layers it wants to receive, and so the network layer should offer subscribe/unsubscribe, or enable/disable calls over a socket descriptor to access specific layers of the data stream.

The send and recv calls expose the packet service to the network layer.

5.4.4 Security and Access Control

Security and access control is obviously a major concern for an Internet architecture, and should therefor be considered seriously. A constrictive security policy lessens the attractiveness of a network, and openness may encourage usage.

This architecture already seems to require security and admission control on two different levels: when an end-host wishes to join a group, and when an end-host wishes to join a group *layer*.

It may be that only one or the other is required, and the more coarsely-grained former option may be enough.

Issues such as shared keys, encryption mechanisms, the actual process of gaining admission to a group, etc, are all extremely important, and have only been hinted at in this report. There are additional issues which may be worthwhile considering: some groups may explicitely benefit from a lack of anonymity, allowing an end-host to be confident of who the recipients of that data will be. In a fuller design, they would be discussed in much greater detail.

5.5 Feasibility and Summary

In this chapter, I have covered various possibilities for offering a group-centric Internet architecture; the possibilities which currently look most suitable for deployment are conceptually a combination of HBH for tree generation, the SSM model of communication for many groups, GLOP for addressing, layered congestion control mechanisms, and how to support these directly within the network. These ideas together form a starting point to explore a new group-centric architecture with a different usage model to that of the existing Deering multicast model.

Currently, the ideas for the new architecture utilise a more structured network locator than exists in IP, with the inter-domain system routing over AS numbers, and fixed-size end-host IDs; the combination of the AS number and the endpoint ID is enough to locate the end-host in the network. The address structure has space for each end-host to be able to create many group, and layering within each group to provide congestion control. At the transport layer, a group ID exists which is distinct from, but mappable to, the network locator which identifies the group owner or coordinator.

The use of a protocol like HBH at the inter-domain level allows not only for a multicast system where forwarding state grows as the multicast group expands, but also one

where the core of the network can avoid duplicating packets entirely. This allows the core of the Internet to remain a simple packet forwarding service, with a little additional state in transit ASes outwith end-sites and end-hosts.

This architecture supports my thesis statement, that a group-centric architecture may be feasible and suitable for modern application deployment, by exploring initial ideas on an architecture which has group communication as its fundamental communication primitive and the various issues of state placement that this generates.

I have presented an initial direction in which to take the work, and possible vectors to test and analyse the architecture and how it might behave in a real-world Internet environment.

Chapter 6

Conclusions & Future Work

Group communication at the network layer could help the application developer and reduce the bandwidth requirements for some applications, aiding service providers in deploying new applications or services. This report has shown not only that current network usage is primarily group-based, but also that there are various flaws in the current Internet architecture, and also in the existing multicast model on which attempts at providing an inter-domain multicast service have been based.

In supporting the thesis statement presented in §1.3, this report has covered the current Internet architecture, various related works, and initial ideas on how to construct an architecture which directly supports group communication. Further work to support this thesis statement will involve devising experiments which verify the feasibility of the architecture, and how it might behave in an inter-domain routing setting.

A desirable property is that the routing and forwarding mechanisms be mathematically sound in terms of worst-case memory/link usage for point-to-point routing between ASes and also for multicast tree maintenance. While this sort of analysis may not be feasible for the multicast distribution mechanism to recipients, there is scope for introducing some of the compact routing work into the routing system used to locate group owners.

Even without theoretical bounds on state placement, any feasible Internet architecture must be able to demonstrate acceptable average-case forwarding table sizes and control traffic volumes, and also acceptable worst-cases for each of these. The key difficulty with this work is in investigating actual behaviour on an Internet-scale, but it should be possible to extrapolate enough information about the scalability of the protocols by engineering various experiments which use realistic traffic patterns over portions of the AS graph.

6.1 Outline Schedule

To fit within a three year PhD term, the required work must be planned. In Fig.6.1 I identify four main phases for the remainder of the work. These are:

- The initial design. This phase will involve determining exactly what needs to be

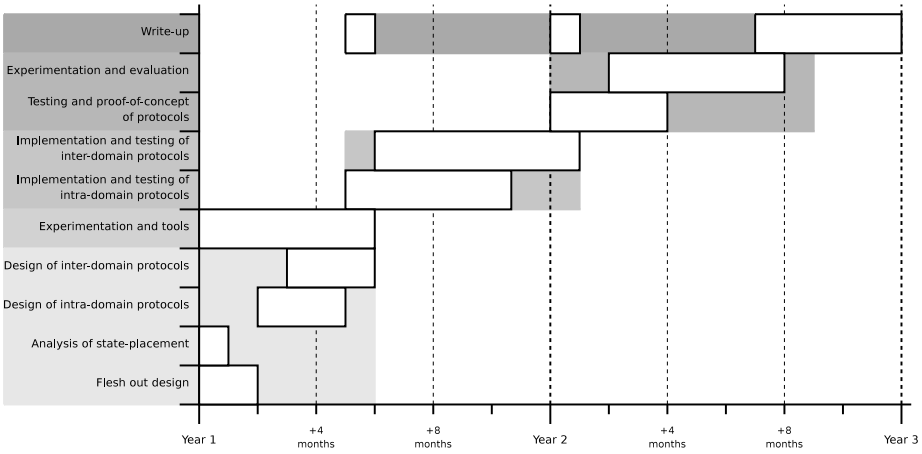


Figure 6.1: PhD plan.

built out of the design to support my thesis statement, in such a way as to narrow the scope of the project and sustain a reasonable amount of work required. To ease the implementation of the key components of such an architecture, the initial phases will include lots of pen & paper investigative work to ensure that the concepts and protocols are sound in so far as can be determined *prior* to serious implementation.

- An experimentation and tools phase, wherein I investigate setting up a testbed network, the XORP platform[1], the Click modular router, simulators such as ns2, network emulation platforms built using Xen virtualisation, etc, for use throughout much of the remainder of the PhD.
- The implementation phase, which picks up these protocols and builds and tests serious implementations of these protocols.
- The investigative phase, which aims to ascertain the viability of the architecture, which should aim to cover: some test applications and how they behave; testing of a small intra-domain setup; testing of a small inter-domain setup with simulated group memberships at the edges. Collection of various data involved, including the state required to maintain various groups of different sizes, the minimum, maximum, and average stretch factors for packets
- The write-up phase, essentially ongoing throughout the PhD, which aims to capture all the various decisions made (including rationale), and all present all relevant data captured.

This appears to be a coherent timetable. If successful, the work will demonstrate a feasible, flexible Internet architecture, and may also help open up further discussion on the usage model we should expect of the future Internet. However, it may be the case that, for example, the HBH tree generating mechanism still requires more forwarding state than might be reasonably handled at an AS, in which case the model may need to be re-evaluated (with, for example, more transport-layer coordination of end-hosts using

[1]The eXtensible Open Router Platform, `http://www.xorp.org/`.

something like Xcast for transient groups). I'd hope to have any major concerns identified and remedied while investigation over experimentation techniques is ongoing, but before serious implementation work has started.

Bibliography

[1] G. Huston, “IPv4 Address Report,” http://ipv4.potaroo.net/, accessed 05/05/2008.

[2] ——, “Growth of the BGP Table - 1994 to Present,” http://bgp.potaroo.net/, accessed 05/05/2008.

[3] R. J. Clark and M. H. Ammar, “Providing scalable Web services using multicast communication,” in *Second International Workshop on Services in Distributed and Networked Environments, 1995*, June 1995, pp. 19–26.

[4] S. Radhakrishnan and C. N. Sekharan, “Multicast Routing with Delay and Delay Variation Constraints for Collaborative Applications on Overlay Networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 3, pp. 421–431, 2007.

[5] G. N. Rouskas and I. Baldine, “Multicast Routing with End-to-End Delay and Delay Variation Constraints,” *IEEE Journal on Selected Areas in Communications*, vol. 15, no. 3, April 1997.

[6] J. H. Saltzer, D. P. Reed, and D. D. Clark, “End-to-end Arguments in System Design,” *ACM Transactions on Computer Systems*, vol. 2, no. 4, pp. 277–288, November 1984.

[7] D. Clark, “The Design Philosophy of the DARPA Internet Protocols,” in *SIGCOMM ’88: Symposium proceedings on Communications architectures and protocols*, August 1988, pp. 106–114.

[8] H. Zimmermann, “OSI Reference Model – The ISO Model of Architecture for Open Systems Interconnection,” *IEEE Transactions on Communications*, vol. 28, no. 4, pp. 425 – 432, April 1988.

[9] B. Carpenter, “Architectural Principles of the Internet,” RFC 1958, June 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1958.txt

[10] D. D. Clark and D. L. Tennenhouse, “Architectural Considerations for a New Generation of Protocols,” in *SIGCOMM ’90: Proceedings of the ACM symposium on Communications architectures & protocols*, September 1990, pp. 200–208.

[11] M. Handley and J. Crowcroft, "Network text editor (NTE): A scalable shared text editor for the MBone," in *SIGCOMM '97: Proceedings of the ACM SIGCOMM '97 conference on Applications, technologies, architectures, and protocols for computer communication*. New York, NY, USA: ACM, 1997, pp. 197–208.

[12] S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne, and L. Zhang, "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 784–803, 1997.

[13] R. Braden, T. Faber, and M. Handley, "From Protocol Stack to Protocol Heap – Role-Based Architecture," *SIGCOMM Computer Communication Review*, vol. 33, no. 1, pp. 17–22, 2003.

[14] J. Saltzer, "On the Naming and Binding of Network Destinations," RFC 1498, August 1993. [Online]. Available: http://www.ietf.org/rfc/rfc1498.txt

[15] D. D. Clark, K. Sollins, J. Wroclawski, and T. Faber, "Addressing Reality: An Architectural Response to Real-World Demands on the Evolving Internet," in *FDNA '03: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, 2003, pp. 247–257.

[16] D. Clark, L. Chapin, V. Cerf, R. Braden, and R. Hobby, "Towards the Future Internet Architecture," RFC 1287, December 1991. [Online]. Available: http://www.ietf.org/rfc/rfc1287.txt

[17] S. Ratnasamy, S. Shenker, and S. McCanne, "Towards an Evolvable Internet Architecture," *SIGCOMM Computer Communication Review*, vol. 35, no. 4, pp. 313–324, 2005.

[18] D. Thaler, M. Talwar, A. Aggarwal, L. Vicisano, and T. Pusateri, "Automatic IP Multicast Without Explicit Tunnels (AMT)," October 2007. [Online]. Available: http://tools.ietf.org/html/draft-ietf-mboned-auto-multicast-08

[19] R. Bush and D. Meyer, "Some Internet Architectural Guidelines and Philosophy," RFC 3439, December 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3439.txt

[20] P. Baran, "On Distributed Communications Networks," *IEEE Transactions on Communications*, vol. 12, no. 1, pp. 1–9, March 1964.

[21] D. W. Davies, K. A. Bartlett, R. A. Scantlebury, and P. T. Wilkinson, "A Digital Communication Network for Computers Giving Rapid Response at Remote Terminals," in *SOSP '67: Proceedings of the first ACM symposium on Operating System Principles*. New York, NY, USA: ACM, October 1967, pp. 2.1–2.17.

[22] J. M. McQuillan and D. C. Walden, "The ARPA Network Design Decisions," *Computer Networks*, vol. 1, pp. 243–289, 1977.

[23] V. G. Cerf and R. E. Kahn, "A Protocol for Packet Network Intercommunication," *SIGCOMM Computer Communication Review*, vol. 22, no. 5, pp. 637–648, 1974.

[24] Information Sciences Institute, University of Southern California, “Internet Protocol Specification,” RFC 791, September 1981. [Online]. Available: http://www.ietf.org/rfc/rfc791.txt

[25] IANA, “Special-Use IPv4 Addresses,” RFC 3330, September 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3330.txt

[26] E. C. Rosen, “Exterior Gateway Protocol (EGP),” RFC 827, October 1982. [Online]. Available: http://www.ietf.org/rfc/rfc827.txt

[27] K. Lougheed and Y. Rekhter, “A Border Gateway Protocol,” RFC 1105, June 1989. [Online]. Available: http://www.ietf.org/rfc/rfc1105.txt

[28] V. Fuller, T. Li, J. Yu, and K. Varadhan, “Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy,” RFC 1519, September 1993. [Online]. Available: http://www.ietf.org/rfc/rfc1519.txt

[29] S. Bradner and A. Mankin, “IP: Next Generation (IPng) White Paper Solicitation,” RFC 1550, December 1993. [Online]. Available: http://www.ietf.org/rfc/rfc1550.txt

[30] S. Bradner, “The Recommendation for the IP Next Generation Protocol,” RFC 1752, January 1995. [Online]. Available: http://www.ietf.org/rfc/rfc1752.txt

[31] I. Castineyra, N. Chiappa, and M. Steenstrup, “The Nimrod Routing Architecture,” RFC 1992, August 1996. [Online]. Available: http://www.ietf.org/rfc/rfc1992.txt

[32] P. Ford and M. Knopper, “TUBA as IPng: A White Paper,” March 1994. [Online]. Available: http://tools.ietf.org/id/draft-ford-ipng-tuba-whitepaper-00.txt

[33] R. Hinden, “Simple Internet Protocol Plus White Paper,” RFC 1710, October 1994. [Online]. Available: http://www.ietf.org/rfc/rfc1710.txt

[34] M. McGovern and R. Ullmann, “CATNIP: Common Architecture for the Internet,” RFC 1707, October 1994. [Online]. Available: http://www.ietf.org/rfc/rfc1707.txt

[35] J. Moy, “OSPF Version 2,” RFC 2328, April 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2328.txt

[36] e. D. Oran, “OSI IS-IS Intra-domain Routing Protocol,” RFC 1142, April 1998. [Online]. Available: http://www.ietf.org/rfc/rfc1142.txt

[37] Y. Rekhter, T. Li, and S. Hares, “A Border Gateway Protocol 4 (BGP-4),” RFC 4271, January 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4271.txt

[38] T. Hardie, “Distributing Authoritative Name Servers via Shared Unicast Addresses,” RFC 3258, April 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3258.txt

[39] S. E. Deering, “Multicast Routing in Internetworks and Extended LANs,” in *SIGCOMM Summer 1988 Proceedings*, August 1988, pp. 55–64.

[40] ——, “Multicast Routing in Datagram Internetworks and Extended LANs,” *ACM Transactions on Computer Systems*, vol. 8, no. 2, pp. 85–110, May 1990.

[41] M. R. Macedonia and D. P. Brutzman, “MBone Provides Audio and Video Across the Internet,” *Computer*, vol. 27, no. 4, pp. 30–36, April 1994.

[42] D. Meyer and P. Lothberg, “GLOP Addressing in 233/8,” RFC 3180, September 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3180.txt

[43] D. Meyer, “Administratively Scoped IP Multicast,” RFC 2365, July 1998. [Online]. Available: http://www.ietf.org/rfc/rfc2365.txt

[44] Z. Albanna, K. Almeroth, D. Meyer, and M. Schipper, “IANA Guidelines for IPv4 Multicast Address Assignments,” RFC 3171, August 2001. [Online]. Available: http://www.ietf.org/rfc/rfc3171.txt

[45] D. Waitzman, C. Partridge, and S. Deering, “Distance Vector Multicast Routing Protocol,” RFC 1075, November 1988. [Online]. Available: http://www.ietf.org/rfc/rfc1075.txt

[46] A. Adams, J. Nicholas, and W. Siadak, “Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised),” RFC 3973, January 2005. [Online]. Available: http://www.ietf.org/rfc/rfc3973.txt

[47] J. Moy, “Multicast Extensions to OSPF,” RFC 1584, March 1994. [Online]. Available: http://www.ietf.org/rfc/rfc1584.txt

[48] T. Ballardie, P. Francis, and J. Crowcroft, “Core Based Trees (CBT),” in *SIGCOMM ’93: Conference proceedings on Communications architectures, protocols and applications*. New York, NY, USA: ACM, 1993, pp. 85–95.

[49] C. Shields and J. J. Garcia-Luna-Aceves, “The Ordered Core Based Tree Protocol,” in *INFOCOM ’97: Proceedings of the INFOCOM ’97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution*. Washington, DC, USA: IEEE Computer Society, 1997, p. 884.

[50] B. Fenner, M. Handley, H. Holbrook, and I. Kouvelas, “Protocol Independent Multicast - Sparse Mode (PIM-SM): Protocol Specification (Revised),” RFC 4601, August 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4601.txt

[51] M. Handley, I. Kouvelas, T. Speakman, and L. Vicisano, “Bidirectional Protocol Independent Multicast (BIDIR-PIM),” RFC 5015, October 2007. [Online]. Available: http://www.ietf.org/rfc/rfc5015.txt

[52] H. W. Holbrook and D. R. Cheriton, “IP Multicast Channels: EXPRESS Support for Large-scale Single-source Applications,” in *SIGCOMM ’99: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*. New York, NY, USA: ACM, 1999, pp. 65–78.

[53] H. Holbrook and B.Cain, “Source-Specific Multicast for IP,” RFC 4607, August 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4607.txt

[54] P. Salova, “Overview of the Internet Multicast Routing Architecture,” RFC 5110, January 2008. [Online]. Available: http://www.ietf.org/rfc/rfc5110.txt

[55] D. Thaler, "Border Gateway Multicast Protocol (BGMP): Protocol Specification," RFC 3913, September 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3913.txt

[56] P. Radoslavov, D. Estrin, R. Govindan, M. Handley, S. Kumar, and D. Thaler, "The Multicast Address-Set Claim (MASC) Protocol," RFC 2909, September 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2909.txt

[57] T. Bates, Y. Rekhter, R. Chandra, and D. Katz, "Multiprotocol Extensions for BGP-4," RFC 2858, June 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2858.txt

[58] B. Fenner and D. Meyer, "Multicast Source Discovery Protocol (MSDP)," RFC 3618, October 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3618.txt

[59] G. Huston, "Analyzing the Internet's BGP Routing Table," *The Internet Protocol Journal*, vol. 4, no. 1, March 2001.

[60] D. Meyer, L. Zhang, and K. Fall, "Report from the IAB Workshop on Routing and Addressing," RFC 4984, September 2007. [Online]. Available: http://www.ietf.org/rfc/rfc4984.txt

[61] M. Wang, A. Goel, and B. Prabhakar, "Tackling IPv6 Address Scalability from the Root," in *IPv6 '07: Proceedings of the ACM SIGCOMM workshop on IPv6 and the Future of the Internet*, August 2007.

[62] D. D. Clark, J. Wroclawski, K. R. Sollins, and R. Braden, "Tussle in Cyberspace: Defining Tomorrow's Internet," in *SIGCOMM '02: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, August 2002, pp. 347–356.

[63] D. Krioukov and kc klaffy, "Toward Compact Interdomain Routing," Unpublished draft, August 2005.

[64] M. Caesar and J. Rexford, "BGP Routing Policies in ISP Networks," *IEEE Network*, vol. 19, no. 6, pp. 5–11, 2005.

[65] L. Kleinrock and F. Kamoun, "Hierarchical Routing for Large Networks," *Computer Networks*, vol. 1, no. 3, pp. 155 – 174, 1977.

[66] D. Thaler and M. Handley, "On the Aggregatability of Multicast Forwarding State," vol. 3, 2000, pp. 1654–1663 vol.3. [Online]. Available: http://ieeexplore.ieee.org/iel4/6725/17999/00832564.pdf

[67] D. Meyer and P. Lothberg, "GLOP Addressing in 233/8," RFC 2770, February 2000. [Online]. Available: http://www.ietf.org/rfc/rfc2770.txt

[68] R. Braden, D. Clark, S. Shenker, and J. Wroclawski, "Developing a Next-Generation Internet Architecture," www.isi.edu/newarch/DOCUMENTS/WhitePaper.pdf, ISI, Tech. Rep., July 2000.

[69] B. Zhang, W. Wang, S. Jamin, D. Massey, and L. Zhang, "Universal IP Multicast Delivery," *Computer Networks*, vol. 50, no. 6, pp. 781–806, May 2006.

[70] S. Ratnasamy, A. Ermolinskiy, and S. Shenker, "Revisiting IP Multicast," in *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, September 2006, pp. 15–26.

[71] T. H. Z. Stoica, I.; Ng, "Reunite: a recursive unicast approach to multicast," *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, pp. 1644–1653 vol.3, 26-30 Mar 2000.

[72] L. H. M. K. Costa, S. Fdida, and O. Duarte, "Hop By Hop Multicast Routing Protocol," *SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 249–259, 2001.

[73] R. Boivie, N. Feldman, Y. Imai, W. Livens, and D. Ooms, "Explicit Multicast (Xcast) Concepts and Options," RFC 5058, November 2007. [Online]. Available: http://www.ietf.org/rfc/rfc5058.txt

[74] Y. Chu, S. G. Rao, and H. Zhang, "A Case for End System Multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1456–1471, October 2002.

[75] S. D. Strowes, "Peer-to-peer audio conferencing," Master's thesis, University of Glasgow, 2005.

[76] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An application level multicast infrastructure," in *Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems (USITS 2001)*, March 2001, pp. 49–60.

[77] S. Birrer, D. Lu, F. E. Bustamante, Y. Qiao, and P. A. Dinda, "FatNemo: Building a Resilient Multi-source Multicast Fat-Tree," in *WCW*, ser. Lecture Notes in Computer Science, vol. 3293. Springer, 2004, pp. 182–196.

[78] J. Liebeherr and T. K. Beam, "HyperCast: A Protocol for Maintaining Multicast Group Members in a Logical Hypercube Topology," in *Proceedings of 1st International Workshop on Networked Group Communication (NGC '99)*, July 1999, pp. 72–89.

[79] Y. Chawathe, "Scattercast: an adaptable broadcast distribution framework," *Multimedia Systems*, vol. 9, no. 1, pp. 104–118, 2003.

[80] R. Moskowitz and P. Nikander, "Host Identity Protocol (HIP) Architecture," RFC 4423, May 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4423.txt

[81] E. Nordmark and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6," February 2008. [Online]. Available: http://tools.ietf.org/id/draft-ietf-shim6-proto-10.txt

[82] M. O'Dell, "GSE - An Alternative Architecture for IPv6," February 1997. [Online]. Available: http://tools.ietf.org/id/draft-ietf-ipngwg-gseaddr-00.txt

[83] D. Farinacci, V. Fuller, D. Oran, and D. Meyer, "Locator/ID Separation Protocol (LISP)," April 2008. [Online]. Available: http://tools.ietf.org/id/draft-farinacci-lisp-07.txt

[84] D. Massey, L. Wang, B. Zhang, and L. Zhang, "A Scalable Routing System Design for Future Internet," in *IPv6'07: Proceedings of the ACM SIGCOMM workshop on IPv6 and the Future of the Internet*, August 2007.

[85] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, "Internet Indirection Infrastructure," *IEEE/ACM Transactions on Networking*, vol. 12, no. 2, pp. 205–218, 2004.

[86] K. Lakshminarayanan, A. Rao, I. Stoica, and S. Shenker, "Flexible and Robust Large Scale Multicast using i3," Computer Science Division (EECS), University of California, Berkeley, California 94720, Tech. Rep. UCG/CSD-02-1187, June 2002.

[87] D. Clark, R. Braden, A. Falk, and V. Pingali, "FARA: Reorganizing the Addressing Architecture," in *FDNA '03: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*. New York, NY, USA: ACM Press, 2003, pp. 313–321.

[88] X. Yang, D. Clark, and A. W. Berger, "NIRA: A New Inter-Domain Routing Architecture." *IEEE/ACM Trans. Netw.*, vol. 15, no. 4, pp. 775 – 788, August 2007.

[89] L. Subramanian, M. Caesar, C. T. Ee, M. Handley, M. Mao, S. Shenker, and I. Stoica, "HLP: A Next Generation Inter-domain Routing Protocol," in *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 2005, pp. 13–24.

[90] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany*, pp. 329–350, November 2001.

[91] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron, "SCRIBE: A large-scale and decentralized application-level multicast infrastructure," *IEEE Journal on Selected Areas in Communications*, 2002.

[92] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth content distribution in a cooperative environment," in *IPTPS'03*, February 2003.

[93] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications," in *Proceedings of the 2001 ACM SIGCOMM Conference*, 2001, pp. 149–160.

[94] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content Addressable Network," Berkeley, CA, Tech. Rep. TR-00-010, 2000.

[95] C. G. Plaxton, R. Rajaraman, and A. W. Richa, "Accessing nearby copies of replicated objects in a distributed environment," in *SPAA '97: Proceedings of the ninth annual ACM symposium on Parallel algorithms and architectures*, 1997, pp. 311–320.

[96] S. Ratnasamy, M. Handley, R. M. Karp, and S. Shenker, "Application-Level Multicast Using Content-Addressable Networks," in *Proceedings of the Third International COST264 Workshop on Networked Group Communication*. Springer-Verlag, 2001, pp. 14–29.

[97] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Tapestry: An infrastructure for fault-tolerant wide-area location and routing," UC Berkeley, Tech. Rep. UCB/CSD-01-1141, April 2001. [Online]. Available: citeseer.ist.psu.edu/zhao01tapestry.html

[98] S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. H. Katz, and J. D. Kubiatowicz, "Bayeux: an architecture for scalable and fault-tolerant wide-area data dissemination," in *Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video*. ACM Press, June 2001, pp. 11–20.

[99] P. Maymounkov and D. Mazières, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," in *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*. London, UK: Springer-Verlag, 2002, pp. 53–65.

[100] M. Hefeeda, A. Habib, D. Xu, B. Bhargava, and B. Botev, "CollectCast: A Peer-to-Peer Service for Media Streaming," *Multimedia Systems*, vol. 11, no. 1, pp. 68 – 81, November 2005.

[101] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "PROMISE: Peer-to-Peer Media Streaming Using CollectCast," in *MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia*. New York, NY, USA: ACM, 2003, pp. 45–54.

[102] M. Caesar, T. Condie, J. Kannan, K. Lakshminarayanan, I. Stoica, and S. Shenker, "ROFL: Routing on Flat Labels," in *SIGCOMM Computer Communication Review*, vol. 36, no. 4, October 2006, pp. 363–374.

[103] M. Caesar, M. Castro, E. B. Nightingale, G. O'Shea, and A. Rowstron, "Virtual Ring Routing: Network Routing Inspired by DHTs," *SIGCOMM Computer Communication Review*, vol. 36, no. 4, pp. 351–362, October 2006.

[104] P. Ganesan, K. Gummadi, and H. Garcia-Molina, "Canon in G Major: Designing DHTs with Hierarchical Structure," in *ICDCS '04: Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*, 2004, pp. 263–272.

[105] D. Krioukov, K. Fall, and X. Yang, "Compact Routing on Internet-Like Graphs," in *INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 1, March 2004, pp. 219–229.

[106] D. Krioukov, kc klaffy, K. Fall, and A. Brady, "On Compact Routing for the Internet," in *SIGCOMM Computer Communication Review*, vol. 37, no. 3, July 2007, pp. 41–52.

[107] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield, "Plutarch: An Argument for Network Pluralism," in *FDNA '03: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*. New York, NY, USA: ACM, 2003, pp. 258–266.

[108] S. Schmid, L. Eggert, M. Brunner, and J. Quittek, "TurfNet: An Architecture for Dynamically Composable Networks." ser. Lecture Notes in Computer Science, vol. 3457. Springer, 2004, pp. 94 – 114.

[109] B. Ford, "Unmanaged internet protocol: taming the edge network management crisis," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 1, pp. 93–98, 2004.

[110] S. Singh and J. Binkley, "The Virtually Invisible Internet," *IEEE Global Internet Symposium, 2007*, pp. 61 – 66, May 2007.

[111] P. F. Ramakrishna, "IPNL: A NAT-Extended Internet Architecture," in *SIGCOMM '01: Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2001, pp. 69–80.

[112] S. Guha and P. Francis, "An End-Middle-End Approach to Connection Establishment," in *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, 2007, pp. 193–204.

[113] S. Guha, Y. Takeda, and P. Francis, "NUTSS: A SIP-based Approach to UDP and TCP Network Connectivity," in *FDNA '04: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*. New York, NY, USA: ACM Press, 2004, pp. 43–48.

[114] M. Walfish, J. Stribling, M. Krohn, H. Balakrishnan, R. Morris, and S. Shenker, "Middleboxes No Longer Considered Harmful," in *OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation*. Berkeley, CA, USA: USENIX Association, 2004, pp. 15–15.

[115] A. Vahdat, M. Dahlin, T. Anderson, and A. Aggarwal, "Active Names: Flexible Location and Transport of Wide-Area Resources," in *Proceedings of the 2nd USENIX symposium on Internet Technologies and Systems*, October 1999.

[116] D. Cheriton and M. Gritter, "TRIAD: A New Next-Generation Internet Architecture," http://www-dsg.stanford.edu/triad/triad.ps.gz, March 2000.

[117] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A Data-Oriented (and Beyond) Network Architecture," in *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM Press, 2007, pp. 181–192.

[118] L. Vicisano, J. Crowcroft, and L. Rizzo, "TCP-like congestion control for layered multicast data transfer," *INFOCOM '98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, pp. 996–1003 vol.3, 29 Mar-2 Apr 1998.

[119] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven Layered Multicast," in *SIGCOMM '96: Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*. New York, NY, USA: ACM, 1996, pp. 117–130.