LECTURE 22
Designing Large Networks

These notes discuss the key ideas that go into the design of large networks. Using the Internet and its network protocol, IP, as an example, we will discuss some important design principles and the techniques that have enabled the Internet to scale to many millions of connected and heterogeneous networks.

The network layer mechanisms described so far in 6.02 require each node (switch) to maintain routing table entries for every node in the network, and send information proportional to the number of nodes in the routing advertisements. This approach simply does not scale beyond a moderate number of nodes. The key to designing ever larger networks is to figure out ways to manage and distribute routing information efficiently: schemes that require per-node state or that end up sending messages through the entire network upon each change (e.g., link failure near the “edge” of the network) don’t scale well. Our scalability goal is to reduce the state maintained by the switches and the bandwidth consumed by control traffic (e.g., routing messages, periodic flooding of packets/messages, etc.) as much as we can.

The fundamental reason for the scalability of any network, including the Internet, is the use of topological addressing. A network addresses (such as an IP address) depends on the network interface’s location in the network topology. Strictly speaking, hosts themselves do not have IP addresses—their network interfaces, also known as network “attachment points,” do. Any given host would have one IP address for each of its currently connected network interfaces, and each of those IP addresses tell the rest of the Internet something about their location.\footnote{For now, we’ll assume that the computer isn’t behind a network address translation box or firewall that provides a private address that is not globally accessible.} Topological addressing allows the routes to IP addresses to be aggregated in the forwarding tables of the routers, and allows routes to be summarized and exchanged by the routers participating in the Internet’s routing protocols. In the absence of aggressive aggregation, there would be no hope for scaling the system to huge numbers; indeed, scalable network routing is challenging enough even with the many techniques that are in use today.

Because an IP address signifies location in the network topology, the Internet’s network layer can use a variant of classical area routing to scalably implement the IP forwarding...
path. We will first describe the concept in abstract terms, and then talk about how it is modified and applied in the Internet.

### 22.1 The area routing idea

The simplest form of classical area routing divides a network layer address into two parts: a fixed-length “area” portion (in the most significant bits, say) and an “intra-area” address. Concatenating these two parts gives the actual network address. For example, one might design a 32-bit area routing scheme with 8 bits for the area and 24 bits for the intra-area address. In this model, forwarding is simple: If a router sees a packet not in its own area, it does a lookup on the “area” portion and forwards the packet on, and conversely. The forwarding tables at the border routers include entries for addresses within a given area, and one entry for every other area. The routers inside an area only maintain information about other addresses within the area and an entry for the area’s border router.

One can extend this idea into a deeper hierarchy by recursively allocating areas at each level, and performing the appropriate recursion on the forwarding path. With this extension, in general, one can define “level 0” of the area routing hierarchy to be each individual router, “level 1” to be a group of routers that share a portion of the address prefix, “level 2” to be a group of level 1 routers, and so on. These levels are defined such that for any level $i$, there is a path between any two routers in the level $i-1$ routers within level $i$ that does not leave level $i$.

There are two reasons why this classical notion of area routing does not directly work in practice:

1. The natural determinant of an area is administrative, but independently administered networks vary widely in size. As a result, it’s usually hard to determine the right size of the “area” field at any level of the hierarchy. A fixed length for this simply does not work.

2. Managing and maintaining a carefully engineered explicit hierarchy tends to be hard in practice, and does not scale well from an administrative standpoint.

### 22.2 Applying area routing to the Internet: Address classes

The second point above suggests that we should avoid a deep hierarchy that requires manual assignment and management. However, we can attempt to overcome the first problem above by allocating network addresses to areas according to the expected size of an area.

When version 4 of IP (“IPv4”) was standardized with 1981 (RFC 791), addresses were standardized to be 32-bits long. At this time, addresses were divided into classes, and organizations could obtain a set of addresses belonging to a class. Depending on the class, the first several bits correspond to the “network” identifier and the others to the “host” identifier. Class A addresses start with a “0”, use the next 7 bits of network id, and the last 24 bits of host id (e.g., MIT has a Class A network, with addresses in dotted-decimal notation of the form 18.*). Class B addresses start with “10” and use the next 14 bits for network id and the last 16 bits for host id. Class C addresses start with “110” and use the
next 21 bits for network id, and the last 8 bits for host id.\(^2\)

Thus, in the original design of IPv4, areas were allocated in three sizes: Class A networks had a large number of addresses, \(2^{24}\), Class B networks had \(2^{16}\) addresses each, and Class C networks had \(2^8\) addresses each.

The router forwarding path in IPv4 with such class-based addressing is quite straightforward: a router determines for an address not in its area which class it belongs to, and performs a fixed-length lookup depending on the class.

### 22.2.1 The problem with class-based addressing

The Internet Engineering Task Force, which is the body that sets the Internet’s standards, realized that this two-level network-host hierarchy would soon eventually prove insufficient and in 1984 added a third hierarchical level corresponding to “subnets”. Subnets can have any length and are specified by a 32-bit network “mask”; to check if an address belongs to a subnet, take the address and zero out all the bits corresponding to zeroes in the mask; the result should be equal to the subnet id for a valid address in that subnet. The only constraint on a valid mask is that the 1s must be contiguous from most significant bit, and all the 0s must be in the least significant bits.

With subnets, the class-based addressing approach served the Internet well for several years, but ran into scaling problems in the early 1990s as the number of connected networks continued growing dramatically. The problem was *address depletion*—available addresses started running out.

It is important to understand that the problem isn’t that the entire space of \(2^{32}\) addresses (about 4 billion) started running out, but that the class-based network address assignment started running out. This is the result of a fundamental inefficiency in the coarse-grained allocation of addresses in the Class A and (often) Class B portions of the address space. The main problem was that Class B addresses were getting exhausted; these were the most sought after because Class A addresses required great explanation to the IANA (Internet Assigned Numbers Authority) because of the large \(2^{24}\) host addresses, which Class C addresses with just 256 hosts per network were grossly inadequate. Because only \(2^{14} = 16384\) Class B networks are possible, they were running out quickly.

### 22.2.2 One solution: CIDR

In a great piece of just-in-time engineering, the IETF stewarded the deployment of CIDR (pronounced “Cider” with a short “e”), or *Classless Inter-Domain Routing*, recognizing that the division of addresses into classes was inefficient and that routing tables were exploding in size because more and more organizations were using non-contiguous Class C addresses.

CIDR optimizes the common case. The common case is that while 256 addresses are insufficient, most organizations require at most a few thousand. Instead of an entire Class B, a few Class C’s will suffice. Furthermore, making these contiguous will reduce routing table sizes because routers aggregate routes based on IP prefixes in a classless manner.

With CIDR, each network gets a portion of the address space defined by two fields,

---

\(^2\)Class D addresses are used for IP multicast; they start with “1110” and use the next 28 bits for the group address. Addresses that start with “1111” are reserved for experiments.
A and \( m \). \( A \) is a 32-bit number (often written in dotted decimal notation) signifying the address space and \( m \) is a number between 1 and 32. \( A \) could be thought of as the prefix and \( m \) the mask: if a network is assigned an address region denoted \( A/m \), it means that it gets the \( 2^{32-m} \) addresses all sharing the first \( m \) bits of \( A \). For example, the network “18.31/18” corresponds to the \( 2^{14} = 16384 \) addresses in the range [18.31.0.0, 18.31.63.255].

### 22.2.3 IP forwarding with CIDR: Longest prefix match

The forwarding step with CIDR can no longer be based on determining the class of an address and doing a fixed-length match. Instead, a router needs to implement a prefix match to check if the address being looked-up falls in the range \( A/m \) for each entry in its forwarding table.

A simple prefix match works when the Internet topology is a tree and there’s only one shortest path between any two networks in the Internet. The topology of the Internet is not a tree, however; many networks multi-home with multiple other networks for redundancy and traffic load balancing (redundancy is the most common reason today).

The consequence of having multiple possible paths is that a router needs to decide on its forwarding path which of potentially several matching prefixes to use for an address being looked-up. By definition, IP (CIDR) defines the correct route as the one corresponding to the longest prefix in the routing table that matches the sought destination address. As a result, each router must implement a longest prefix match (LPM) algorithm on its forwarding path.