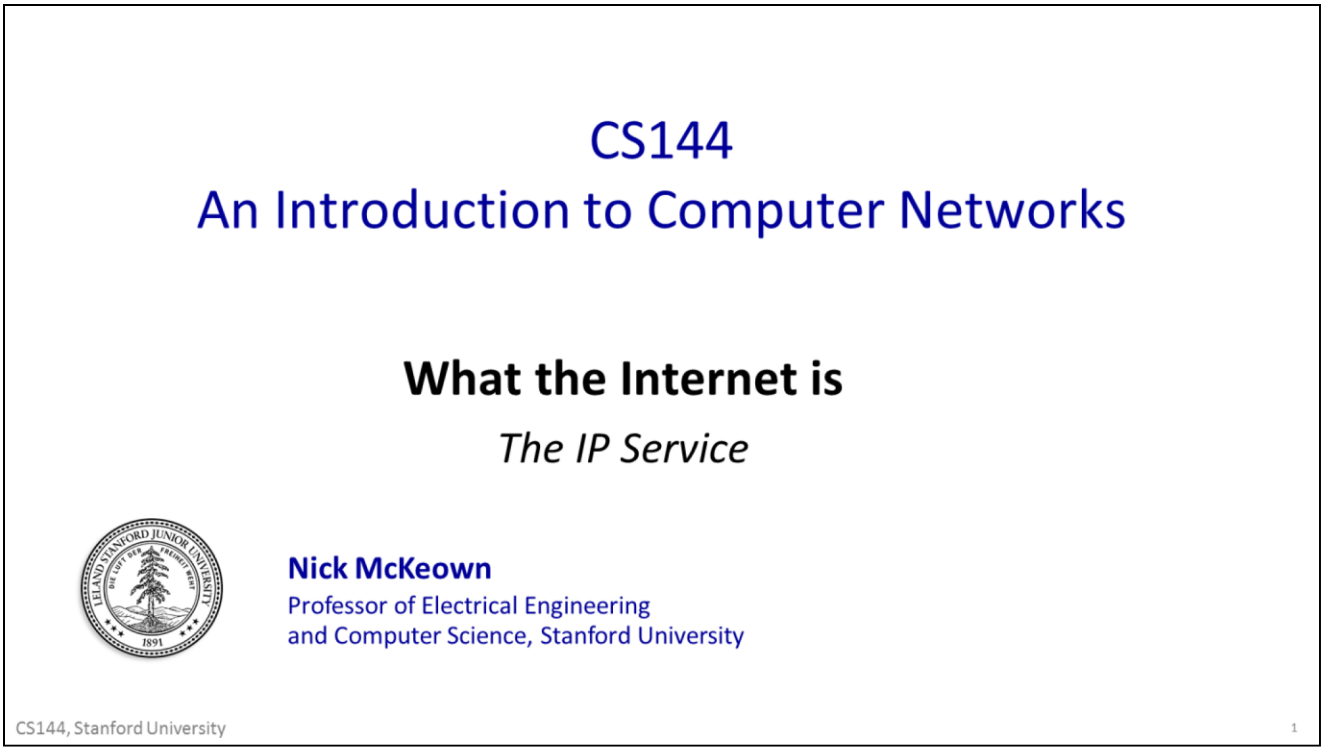


CS144
An Introduction to Computer Networks

What the Internet is

The IP Service

Nick McKeown
Professor of Electrical Engineering
and Computer Science, Stanford University

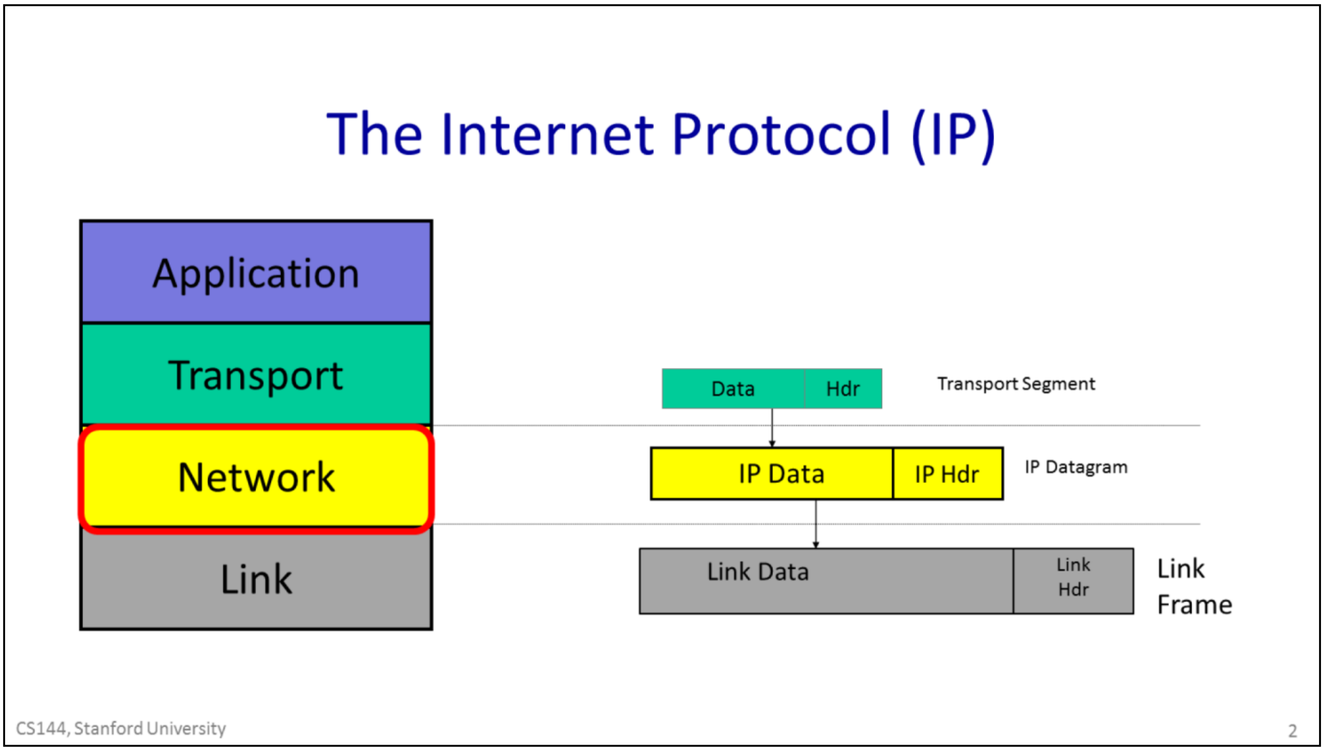
CS144, Stanford University 1

Now that you have learned about the 4-Layer Internet model, we are going to focus on the Network Layer.

This is the most important layer of the Internet – in fact, to many people it *is* the Internet. Whenever we use the Internet we are required to use the Internet Protocol to send and receive packets.

You’ll remember that we say each layer provides a “service” to the layer above. In order to correctly use a layer, we need a good understanding of the service it provides.

Therefore, in this video I will walk through the service provided by the Internet Protocol.



The Internet Protocol (IP)

Application

Transport

Network

Link

Data | Hdr — Transport Segment

IP Data | IP Hdr — IP Datagram

Link Data | Link Hdr — Link Frame

CS144, Stanford University

2

<click>

IP datagrams consist of a header and some data.

<click> When the transport layer has data to send, it hands a Transport Segment to the Network layer below.

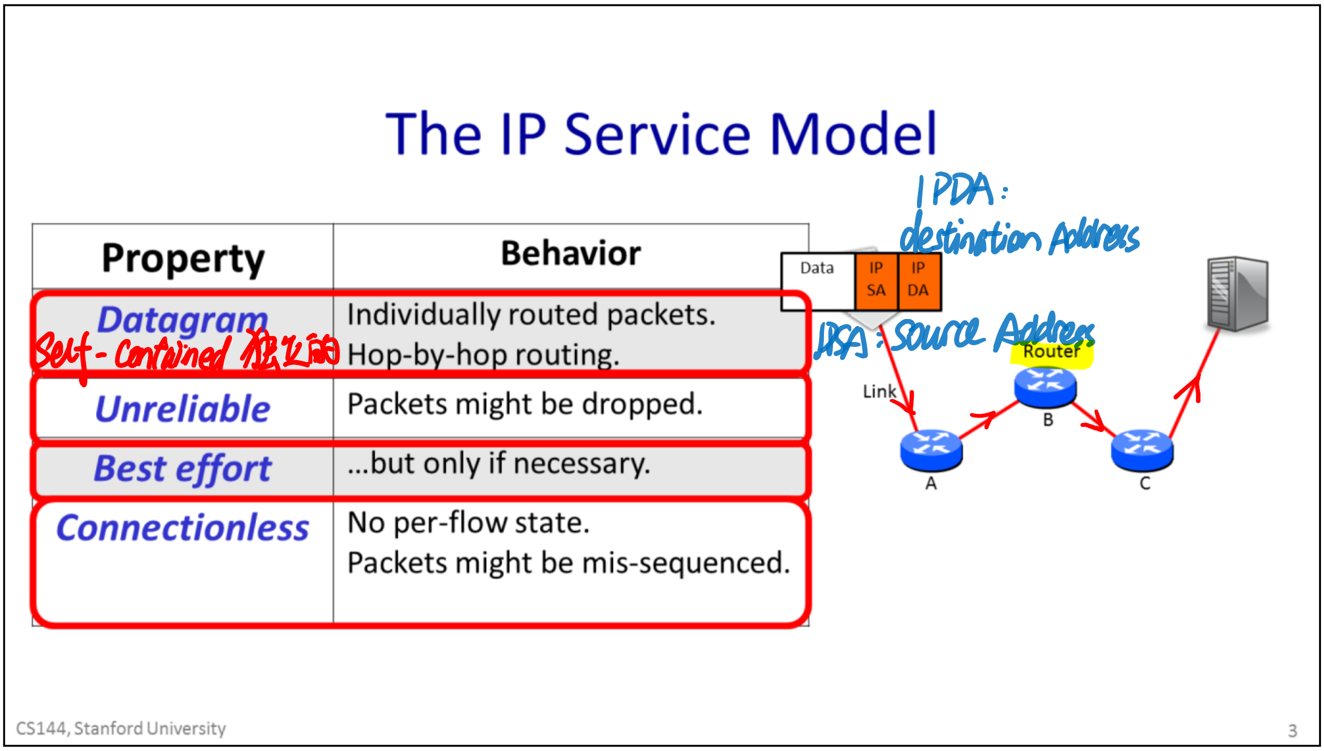
<click to drop transport segment into IP datagram>

The network layer puts the transport segement inside a new IP datagram. IP’s job is to deliver the datagram to the other end.

But first, the IP datagram has to make it over the first link to the first router.

<click to put IP datagram inside Link frame>

IP sends the datagram to the Link Layer that puts it inside a Link frame, such as an Ethernet packet and ships it off to the first router.



The IP Service Model

Property	Behavior
Datagram Self-Contained 独立的	Individually routed packets. Hop-by-hop routing.
Unreliable	Packets might be dropped.
Best effort	…but only if necessary.
Connectionless	No per-flow state. Packets might be mis-sequenced.

IPDA: destination Address

IPSA: Source Address

The IP service can be characterized by four properties listed here. It sends Datagrams from end host to end host; it is unreliable, but makes a best-effort to deliver the datagrams. The network maintains no per-flow state associated with the datagrams.

Let's take a look at each one in turn as listed in the table……

<Click to highlight Datagram> First, IP is a datagram service. When we ask IP to send some data for us, it creates a datagram and puts our data inside. The datagram is a packet that is routed individually through

the network based on the information in its header. In other words, the datagram is self-contained.

<CLICK to make packet appear>The header contains the IP address of the destination, which we abbreviate here as “IP DA” for IP destination address. The forwarding decision at each router is based on the IP DA. The datagram header also contains an IP source address, or “IP SA”, saying where the packet came from, so the receiver knows where to send any response. <Click to make datagram move hop by hop> Datagrams are routed hop-by-hop through the network from one router to the next, all the way from the IP source address to the IP destination address . We’ll learn more about how routers work later. But for now, it’s enough to know that each router contains a forwarding table that tells it where to send packets matching a given destination address. The router doesn’t know the whole path – it simply uses the destination address to index into its forwarding table so that it can forward the packet to the next hop along the path towards its final destination. Hop by hop, step by step the packet makes its way from the source to the destination using only the destination address in the datagram.

You will often hear the analogy made between how IP datagrams are routed and how letters are routed by the postal service. It’s a good analogy. In the postal service, we put a letter into the mail box with the address of the destination and the letter is routed – invisibly to us – hop by hop from sorting office to sorting office until it reaches its destination. Neither the sender or the receiver know – or need to know –

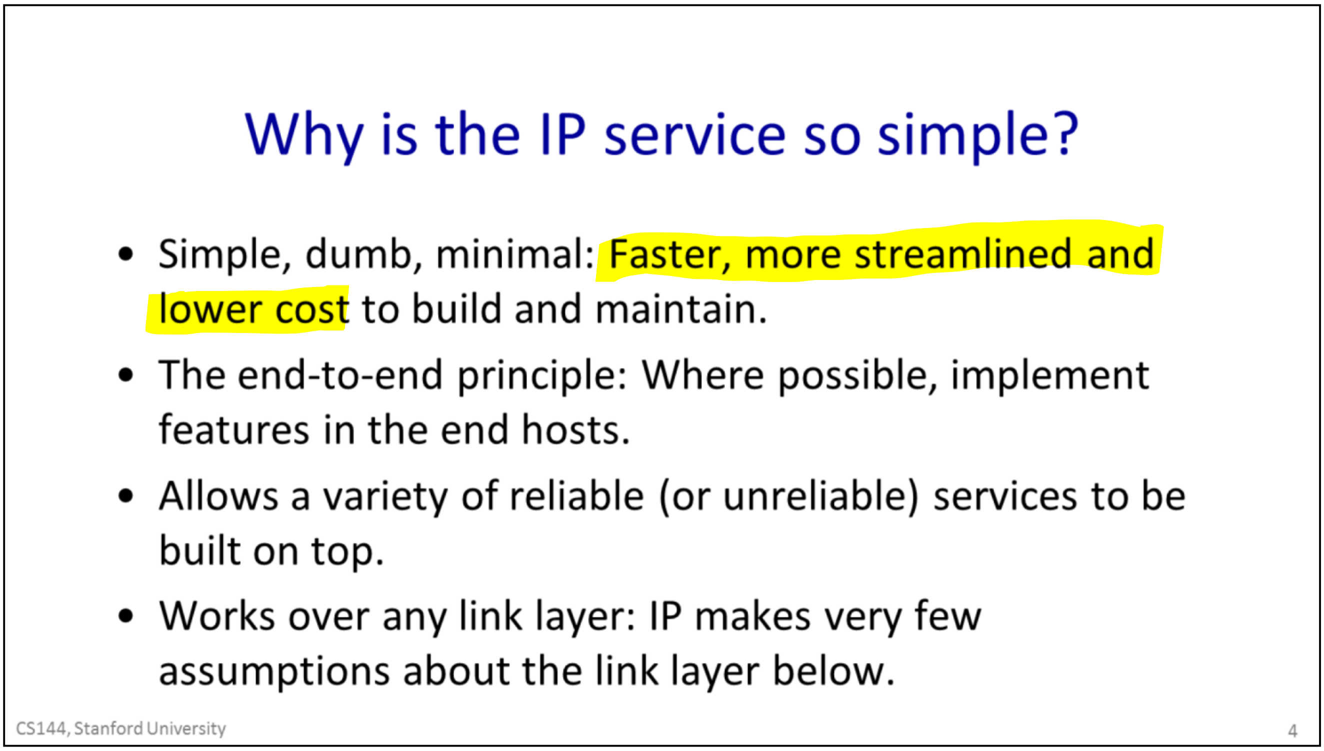
the path taken by letters in the postal service or by datagrams in the Internet. The IP service model provides a service which includes the routing to the destination.

<click to highlight Unreliable> Second, and perhaps surprisingly, IP is unreliable. IP makes no promise that packets will be delivered to the destination. They could be delivered late, out of sequence, or never delivered at all. It’s possible that a packet will be duplicated along the way, for example by a misbehaving router. The key thing to remember is that IP is unreliable and makes no guarantees.

<click to highlight Best Effort> But it won’t drop datagrams arbitrarily just because it feels like it. That’s if you believe networks have feelings. IP does make the promise to only drop datagrams if necessary. For example, the packet queue in a router might fill up because of congestion, forcing the router to drop the next arriving packet. IP won’t make any attempt to resend the data – in fact, IP doesn’t tell the source that the packet was dropped. Similarly, a faulty routing table might cause a packet to be sent to the wrong destination. Or cause a packet to be duplicated by mistake. IP doesn’t makes no promises these errors won’t happen, nor does it detect them when they do. But IP does make the promise to only make these errors when necessary.

In fact, the IP datagram service is very much like the basic postal service. The basic postal service makes no promise that our letters will be delivered on time, or that if we send 2-3 letters on successive days that they will be received in the order they were sent, and it makes no promise they will be delivered at all (unless we pay for a more expensive end-to-end service to guarantee delivery).

Really, when it comes down to it, IP is an extremely simple, minimal service. It maintains no state at all related to a communication. We say that a communication service is “connectionless” <click to highlight connectionless> because it doesn’t start by establishing some end to state associated with the communication. In other words, when we make a Skype call lasting several minutes and consisting of many IP datagrams, the IP layer maintains no knowledge of the call, and simply routes each datagram individually and independently of all the others.



Why is the IP service so simple?

- Simple, dumb, minimal: Faster, more streamlined and lower cost to build and maintain.
- The end-to-end principle: Where possible, implement features in the end hosts.
- Allows a variety of reliable (or unreliable) services to be built on top.
- Works over any link layer: IP makes very few assumptions about the link layer below.

You might be wondering why the IP service is so simple. After all, it is the foundation of the entire Internet. Every communication over the Internet uses – must use – the IP service. Given how important the Internet is, wouldn’t it have been better to make IP reliable? After all, we did say that most applications want a reliable, byte-communication service.

There are several reasons the IP service model was designed to be so simple.

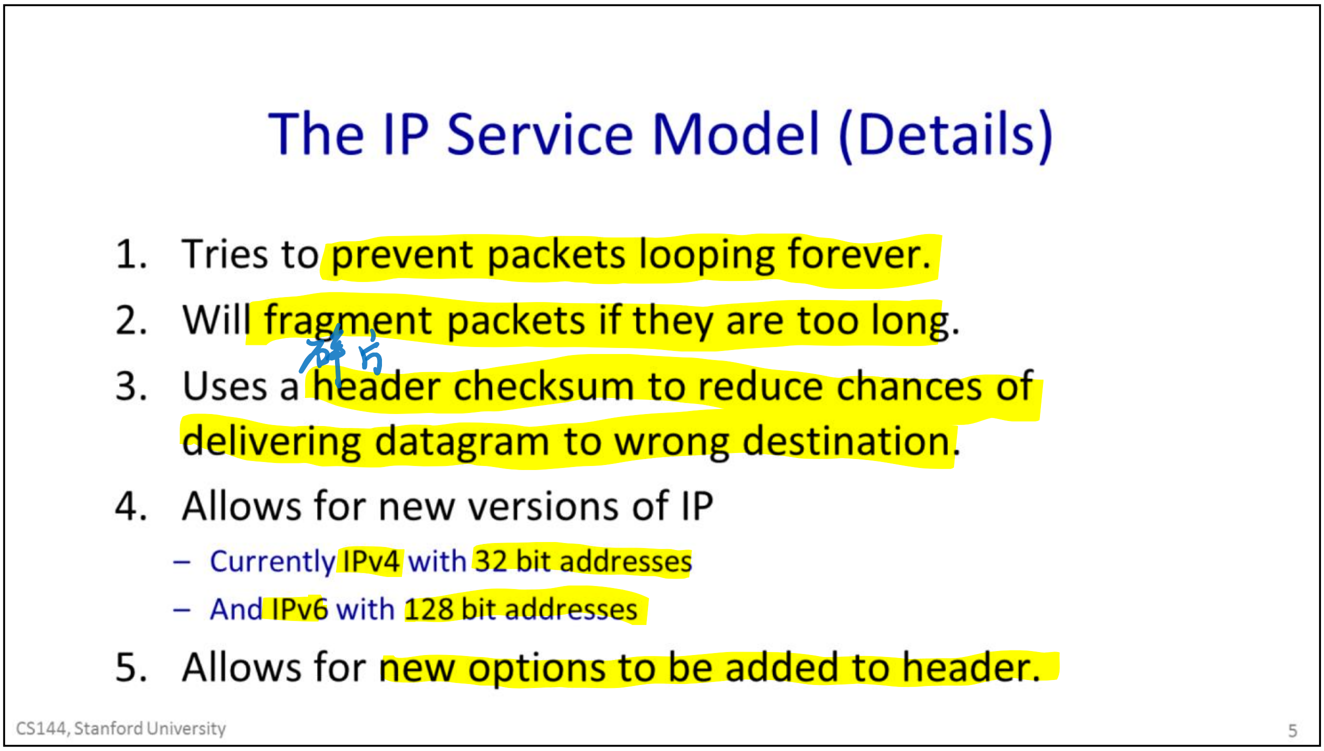
1.<click>To keep the network simple, dumb and minimal. Faster, more streamlined and lower cost to build and maintain. It was believed that if the network is kept simple with very features and requirements, then packets could be delivered very quickly, and at low cost. The thinking was that a simple network could be made to run very fast using dedicated hardware. And given that the network is implemented by a large number of routers scattered throughout the network, if they could be kept simple then are likely to be more reliable, more affordable to maintain and will need to be upgraded less often.

2. <click>The end to end principle: Where possible, implement features in the end hosts. In the design of communication systems, there is a well known principle called the end-to-end principle that says that if you *can*correctly implement features at the end points then you should. We’ll study the end-to-end principle in more depth in later videos, but the basic idea is to place as much intelligence as possible at the end points – in our case, the source and destination computers. This can have several advantages, such as making sure the feature is implemented correctly for the application, and it is easier to evolve and improve a feature if it is implemented in software on end computers rather than baked into the hardware of the Internet. In the case of the Internet, it was decided that features such as reliable communications and controlling congestion should be done at the end points – by the source and destination computers, and not by the network. At the time, it was quite a radical suggestion and a very different design choice from the telephone system, which was originally built on the idea of simple handsets and a complicated feature-rich network of telephone switches. In later

videos we will be studying the end-to-end principle as one of the important architectural principles of communication systems. We will see many examples of the end to end principle in action. For example, when we study the transport layer, we will see how the end hosts build a reliable communication service over the unreliable IP network service.

3. <click>Allows a variety of reliable (or unreliable) services to be built on top. If IP was reliable – in other words if any missing packets were retransmitted automatically – then it would not be ideal for some services. For example, in real time applications like a video chat, there might be no point in retransmitting lost data, because it might arrive too late to be useful. Instead, the application might choose to show a few blank pixels or use the pixels from the frame before. By not providing any reliability guarantees, IP lets the application choose the reliability service its needs.

4.<click>Works over any link layer: IP makes very few assumptions about the link layer. IP makes very little expectation of the Link layer below – the link could be wired or wireless, and requires no retransmission or control of congestion. Some people have said IP is so simple and makes so few assumptions about the underlying link layer that you could run IP over carrier pigeons. In fact, there is even an Internet standard telling you how to do it! Making IP run over any link layer made sense because the Internet was created specifically to interconnect existing networks (which is why it was called the Internet).



The IP Service Model (Details)

1. Tries to prevent packets looping forever.
2. Will fragment packets if they are too long.

碎片

3. Uses a header checksum to reduce chances of delivering datagram to wrong destination.
4. Allows for new versions of IP
   - Currently IPv4 with 32 bit addresses
   - And IPv6 with 128 bit addresses
5. Allows for new options to be added to header.

CS144, Stanford University 5

In addition to the basic unreliable, best-effort, connectionless datagram service, IP also provides a few other carefully chosen services. The designers of IP tried very hard to find a balance between providing the bare minimum needed to make communication work, while not providing such a barebone service that it doesn’t really work.

I’ll describe five features here and you will learn about each one of these features in later videos, so I won’t go into a lot of the details here. But I will briefly describe each one, so you understand the scope of the complete IP service.

<Click> First, IP tries to prevent packets from looping forever. Because IP routers forward packets hop-by-hop across the Internet, it is possible for the forwarding table in a router to be wrong, causing a packet to start looping round and around following the same path. This is most likely to happen when the forwarding tables are changing and they temporarily get into an inconsistent state. Rather than try to prevent loops from ever happening – which would take a lot of complexity - IP uses a very simple mechanism to catch and then delete packets that appear to be stuck in a loop. To do this, IP simply adds a hop-count field in the header of every datagram. It is called the time to live, or TTL field. It starts out at a number like 128 and then is decremented by every router it passes through. If it reaches zero, IP concludes that it must be stuck in a loop and the router drops the datagram. It is a simple mechanism, typical of IP – it doesn't guarantee loops won't happen, it just tries to limit the damage caused by a flood of endlessly looping packets in the network.

<click> IP will fragment packets if they are too long. IP is designed to run over any kind of link. Most links have a limit on the size of the packets they can carry. For example, Ethernet can only carry packets shorter than 1500bytes long. If an application has more than 1500bytes to send, it has to be broken into 1500 pieces before sending in an IP datagram. Now, along the path towards the destination, a 1500byte datagram might need to go over a link that can only carry smaller packets, for example 1000 bytes. The router connecting the two links will fragment the datagram into two

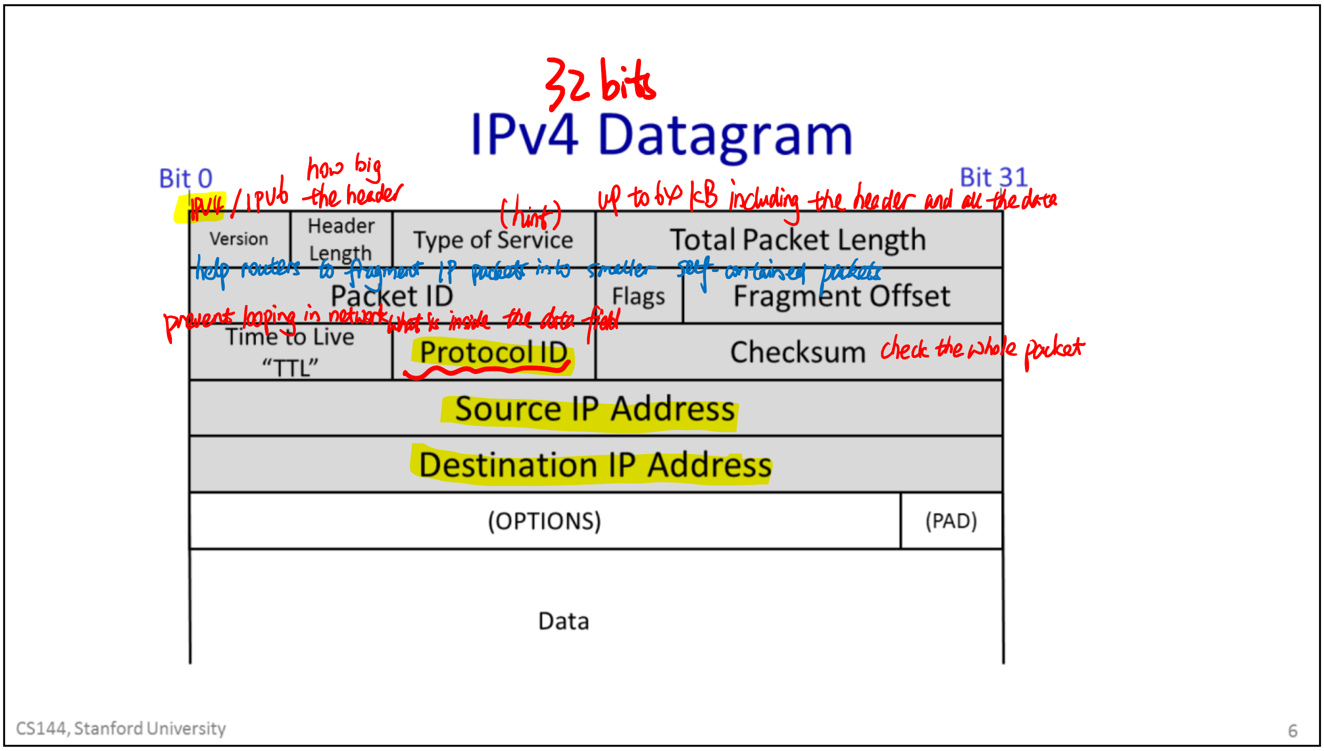
smaller datagrams. IP provides some header fields that we will see in a minute to help the router fragment the datagram into two self-contained IP datagrams, while providing the information the end host needs to correctly reassemble the data again.

<click> IP uses a header checksum to reduce chances of delivering a datagram to the wrong destination. IP includes a checksum field in the datagram header to try and make sure datagrams are delivered to the right location. It could be quite a security problem if packets are accidentally and frequently sent to the wrong place because of a mistake by a router along the way.

<click> There are two versions of IP in use today: IPv4, which is used today by over 90% of end hosts. It uses the 32bit addresses you are probably familiar with. Because we are running out of IPv4 addresses, the Internet is in a gradual transition to IPv6, which uses 128 bit addresses instead. You’ll be learning about the details of IPv4 and IPv6 in later videos.

<click> Finally, IP allows new fields to be added to the datagram header. This is a mixed blessing. On the one hand, it allows new features to be added to the header that turn out to be important, but weren’t in the original standard. On the other hand, these fields need processing and so require extra features in the routers along the path, breaking the goal of a

simple, dumb, minimal forwarding path. In practice, very few options are used or processed by the routers.



32 bits

IPv4 Datagram

Bit 0 … Bit 31

Version — IPv4 / IPv6

Header Length — how big the header

Type of Service — (hint)

Total Packet Length — up to 64 kB including the header and all the data

help routers to fragment IP packets into smaller self-contained packets

Packet ID | Flags | Fragment Offset

Time to Live "TTL" — prevent looping in network

Protocol ID — what is inside the data field

Checksum — check the whole packet

Source IP Address

Destination IP Address

(OPTIONS) | (PAD)

Data

CS144, Stanford University

6

I'm now going to show you the IPv4 header and explain what all the fields do. I don't need you to remember where all the fields are (I don't remember all their locations myself). But I do want you to know what each field does, because it helps you understand the scope of the IP service model. It should help cement your understanding, and make it really clear that IP doesn't do a lot – it is a deliberately simple service.

Here is a picture of an IPv4 header, which is the most common header in use today. I've drawn it here in 32 bit words, with "Bit 0" the first to be sent onto the wire.

This shaded portion is the IPv4 header. It's followed by data.

The most important fields in the IP header are:

1.<click> The Destination IP address

2.<click> The Source IP address

3.<click> The Protocol ID, that tells us what is inside the data field. Essentially, it allows the destination end host to demultiplex arriving packets, sending them to the correct code to process the packet. If the Protocol ID has the value "6" then it tells us the data contains a TCP Segment, and so we can safely pass the datagram to the TCP code and it will be able to parse the segment correctly. The Internet Assigned Numbers Authority (IANA) defines over 140 different values of Protocol ID, representing different transport protocols.

<click> The Version tells us which version of IP – currently, the legal values are IPv4 and IPv6. This header is an IPv4 header. We'll see IPv6 headers in a later video.

<click> The Total packet length can be up 64kBytes including the header and all the data.

<click> The "Time to Live" field helps us to prevent packets accidentally looping in the ntwork forever.

Every router is required to decrement the TTL field. If it reaches zero, the router should drop the packet.

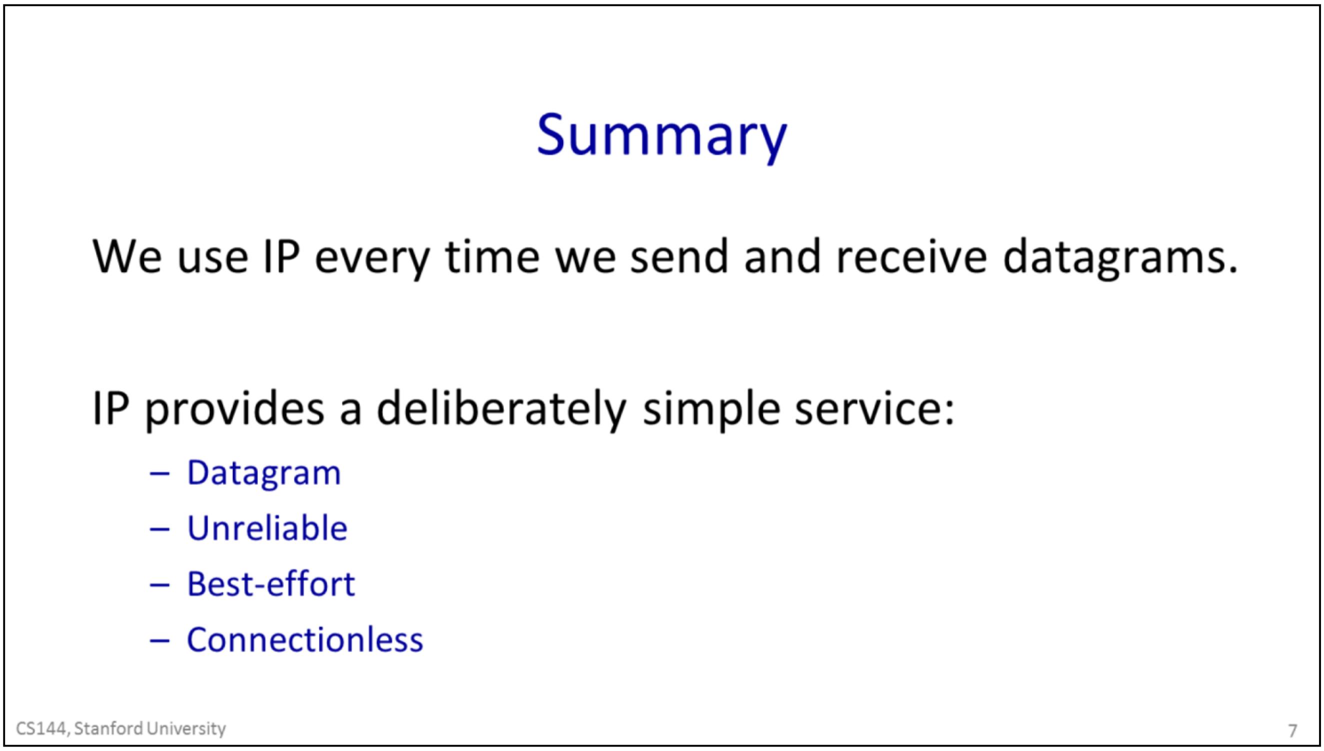
This way, when the source sends the packet with a fixed TTL value, it is guaranteed to be destroyed by a router if it starts to travel in loops.

<click> Sometimes a packet is too long for the link it is about to be sent on. The Packet ID, Flags and Fragment Offset all help routers to fragment IP packets into smaller self-contained packets if need-be. We will learn how fragmentation works in a later video.

<click> The Type of Service field gives a hint to routers about how important this packet is.

<click> The Header Length tells us how big the header is --- some headers have optional extra fields to carry extra information.

<click> Finally, a checksum is calculated over the whole header so just in case the header is corrupted, we are not likely to deliver a packet to the wrong desination by mistake.



Summary

We use IP every time we send and receive datagrams.

IP provides a deliberately simple service:

- Datagram
- Unreliable
- Best-effort
- Connectionless

CS144, Stanford University 7

In summary, IP is very important: <click> We use it every time we send and receive packets in the Internet.

<click> IP provides a deliberately simple service. It is a simple, dumb, minimal service with four main features: It sends datagrams, hop-by-hop across the Internet. The service is unreliable and best-effort; there is no per-flow state making the protocol connectionless.

At this point, you should feel comfortable with what the IP protocol is, what its service model is, and how it fits into the Internet 4-layer hierarchy. If you have doubts, I suggest you re-watch this video and the one before it on the 4-layer model.

You’ll also find lots of good references about how IPv4 works. Any good networking textbook will devote considerable space to explaining what IP is, and why it was designed this way. For example, Chapter 4 of the 6th Edition of “Computer Networking: A top down approach” by Kurose and Ross. You will also find a brief explanation on Wikipedia.

<The End>