# Internet Protocols Hellspawn Document

Jacqueline Walker

May 17, 2024

# 1 Link & Network Communication

## 1.1 Switching Techniques

**Circuit Switching** Route is reserved before sending a message from source to destination. Requires set-up and tear-down time. Best for long messages or telephony.

**Message Switching** One message is sent on the route from node to node until it reaches the destination. Not used frequently because of the need of a large buffer at each node, and large delay in transmission.

**Packet Switching** Message is split up into small packets that are then sent on the route from node to node. Packets may be received out-of-order, but has a significant reduction in delay and buffer size to other methods. Used for network communications i.e. the Internet.

## 1.2 Protocol Stack

Protocols are separated into well-defined layers which interact via interfaces between layers. Each protocol in a higher layer is encapsulated by a lower-level protocol as it descends down the stack. Splitting the process into layers allows each program to conform to a simple, well-defined interface that can slot with any other potential protocol.

ISO-OSI model:

**Application** Program user interacts with.

**Presentation** Provides common interface for requesting services. Abstracts differences in presentation.

**Session** Responsible for setting up and tearing down connections. Abstracts packet communication into one 'stream'.

**Transport** Responsible for splitting packets up for transmission, flow control, and reliable communication (if TCP).

**Network** Work out route needing to be took by each packet across the network. May further split up packets.

**Data Link** Moves one chunk of data from one node to the next.

**Physical** Physical transmission of bits across a link of a certain medium.

## 1.3 Flow Control

Flow control is responsible for ensuring that the receiver is not overwhelemed by the amount of messages being sent by the transmitter, and potentially having its buffer overflow. Ideally, the flow control should aim to keep utilization and throughput as high as possible.

**Software Flow Control** Transmitter sends a stream of packets to the receiver until an XOFF is replied. Transmitter then must await an XON before restarting transmission. Very efficient flow control, but suffers from potential errors if XOFF/XON packets are corrupted or if transmitter and receiver are sufficiently far away.

**Stop-and-Wait** Transmitter sends one packet at a time and awaits an ACK before sending the next packet. Suffers from poor utilization over non-short links. Used for channels with a large propogation distance or high noise.

**Sliding Window** Transmitter sends N packets. Receiver sends and ACK with sequence number for the packet + 1. If the transmitter has received an ACK, it checks if the ACK + N is greater than the transmitters most recent sequence number. If it is, the transmitter may send packets until the difference is 0, or until a new ACK arrives with a greater sequence number + N. Else, the transmitter must wait until an ACK arrives with a greater sequence number + N. Used by TCP.

$$\text{Utilization} = \frac{t_{packet}}{t_{packet} 2t_{prop}}, \text{or}$$

$$U = \frac{1}{1+2a}, \text{where } a = \frac{t_{prop}}{t_{packet}}$$

$$U_{sliding} = \frac{N}{1+2a}$$

Derivation of utilization:

$$\text{Utilization} = \frac{\text{Time spent sending data}}{\text{Total time}}$$

$$U = \frac{t_{packet}}{t_{packet} + 2t_{propogation} + t_{ack} + t_{tx\_processing} + t_{rx\_processing}}$$

Assume that $t_{ack}, t_{tx\_processing}, t_{rx\_processing}$ is negligible.

## 1.4 Error Control

**Stop-and-Wait ARQ** Tx sends one packet and waits for ACK to be received. if no ACK received in $2t_{prop}$, then retransmit. $U = \frac{1}{1+2a} \times$ proportion of packets w/ no error, p

**Go-Back-N ARQ** Tx sends window size of packets before receiving ACK with sequence number for first packet. If packet is not received, Rx sends ACK for that packet continuously until the Tx sends that packet, then carries on from that point. $U = 1 - p(1 + 2a)$

**Selective Repeat** Similar to Go-Back-N, but only retransmits lost packet when NAK for that packet is sent by Rx. $U_{non-continuous} = \frac{N(1-p)}{1+2a}$, $U_{continuous} = 1 - p$

### 1.5 Error Detection

**Parity Bit** Add extra bit(s) that makes the no. of ones/zeros in the message either even or odd. Detects any odd no. of bit errors. Easy to implement in hardware. Used in simple serial communications.

**Checksum** Use algorithm to add up all bits in message and convert to a block of bits to append to message. Detects all single bit errors and most error bursts. Easy to implement in software. Used by IP

**Cyclic Redundancy Check** The message is divided by a given pattern and the resulting number is appended as the FCS. Is much more effective than checksums and is easy to implement in hardware using XOR gates. Used by MAC.

Internet Checksum is computed by adding up each 16-bit word of the IP header together, padding the LSBs with zeros to fit. Any carry bit is added to the LSB. Invert the result.

### 1.6 Multiple Access

**ALOHA** Contention-based. Send any message at any time and wait for ACK. If no ACK is received, resend message. If collision occurs, packets are lost.

**Slotted ALOHA** ALOHA but with TDMA. Increased throughput and reduced collisions from ALOHA. Requires accurate synchronization.

**Carrier Sense Multiple Access** ALOHA but listens to the link for any currently ongoing transmissions and waits if so. Difficult to implement on radio networks, where not all nodes may be able to listen.

**CSMA/CD** Collision detection with CSMA. Listen for any other transmissions when you are transmitting. If collision happened (amplitude greater than expected for single signal), stop transmission.

**CSMA/CA** Collision avoidance for CSMA. Send a short packet to 'reserve' a link for transmitting the message before starting. More efficient than CSMA/CA, and used in wireless LAN (WLAN).

**Polling Based** Contention free. Master node polls each slave node to check if a message needs to be sent. If so, allocates a time slot for transmission. High latency.

**Token Passing** Control token passed between nodes. If node has token it may transmit messages. Lost tokens must be regenerated.

### 1.7 Hubs, Switches & Bridges

Hubs are simple and cheaper devices that pass frames from one port to all ports on the hub. All nodes can then listen to any frames on the network.

Switches are more complex and expensive devices that contains an internal buffer. A frame with only one destination address may be passed directly to the destination port only, giving better security. This also allows for a greater network capacity, as a larger network with a hub would constantly suffer collisions. Switches may also store one frame in its internal buffer to prevent a collision if two frames are received at the same time. A switch acts as a multi-port bridge.

A bridge connects two LANs together, either local or remote. They may also operate using different protocols. Bridges allow two separate LANs to transmit simultaneously within each other, while still allowing communication between.If one side of the bridge fails, the other side stays up. Only packets for the far side of the LAN go through the bridge, so security remains high.

Bridges receive all frames from both side of the LAN and determine whether they need transmission across depending on the destination address. This may be done manually by the network admin or automatically via taking note of which source address is on which port. Loops need to be removed from the network as otherwise packets will be forwarded forever.

### 1.8 Spanning Tree Protocol

The spanning tree protocol is used to create a tree of the entire LAN, where all nodes are accessible without including any loops. It does not guarantee good performance, and if a bridge fails it can reconfigure the network.

1. Each bridge generates a random bridge number, and gives each of its ports a port number and an associated cost, set to zero.
2. All bridges send out their bridge number on all ports with associated cost.
3. A bridge that receives a message will keep note of the lowest bridge number received. If the message does have the lowest number will then start to send out that number instead of its own, with the cost set to the propogation time from the current bridge to the bridge with the lowest number.
4. After reasonable time, all bridges will be sending out the same number and will all have associated costs.
5. If a bridge receives a message with the same bridge number but a lower cost, it will stop broadcasting as there is a better route.

## 2 Protocols

### 2.1 Ethernet

Ethernet is used for wired LANs and standards usually follow the structure (Speed in Mbit/s)BASE(Max wire length in 100m). For example, 10BASE5 denotes an Ethernet that can ransmit at 10Mbit/s for 500m on one wire. Other examples are 10BASE-T, which transmit at 10Mbit/s @ 100m using a twisted pair.

Ethernet used CSMA/CD to detect collisions. It uses an exponential random backoff for determining how long to wait after detecting a collision.

- After a frame has been sent, wait atleast 96 bit times before new frame.

- If collision detected during transmission, wait 512 bit times before halting transmission so all stations can detect the collision
- Wait random time between 0 and $512 \times 2^{min(\text{no. of collisions},10)}$
- Give up after 16 collisions
- First successful station to transmit after collision resets no. of collisions and gets priority in retransmission.

$$\text{min. frame length}, F_{min} \geq 2t_{prop} \times \text{bitrate}$$

Ethernet (MAC) frame format:

**Preamble** 7 bytes alternating 1 0

**Starting Delimiter** 10101011

**Destination/Source** 6 bytes respectively MAC addresses

**Length/Type** 2 bytes, shared because of competing standards of DIX and 802.3. If $\leq 1500$ is length, else protocol of data. If 0x8100, VLAN extension used. If 0x0800, IPv4, if 0x86DD, IPv6.

**Data** 46-1500 bytes. Anything.

**FCS** 4 bytes. CRC result for Ethernet frame.

VLAN extension allows you to divide one physical LAN into multiple 'virtual LANs', which can allow remote stations to stay in the same LAN, and some local stations to be on a separate LAN.

**Length/Type** 2 bytes, 0x8100

**Priority** 3 bits.

**Canonical Format Indicator (CFI)** 1 bit, if set reverse bits to send over Ethernet.

**VLAN ID** 12 bits, used to determine which VLAN frame is for.

**Type** 2 bytes, same as original Length/Type field.

## 2.2 MAC & LLC

Mac addresses are 6 bytes written in hexadecimal. The first 3 bytes are allocated for use to you by the IEEE. THe last 3 bytes are user defined. Ethernet sends LSB first, so bit order is flipped for each byte.

**Group/Individual** 1 bit, broadcast devices or individual only

**Local/Global** 1 bit, if assigned by IEEE or local only

**OUI** Rest of 3 bytes, allocated by IEEE

LLC format (stored in MAC Data field):

**DSAP/SSAP** 1 byte each, used for identifying packets with different higher-layer protocols(?).

**CTL** 1 byte, control field, used to determine what the LLC data is specifed for or where it needs to go.

**Data** Rest of frame data.

SNAP SAP bodge replaces DSAP/SSAP with AA AA. This was necessary as DSAP/SSAP was too small to actually use in practise. This prompts the MAC controller to look at the first 5 bytes of LLC data for the actual protocol used.

LLC Types:

**Type 1** 1 byte CTL, connectionless, best-effort

**Type 2** 1-2 bytes CTL (if using sequence numbers), connection-oriented, reliable with sliding window flow control with N=127

**Type 3** 1 byte CTL, connectionless, reliable

## 2.3 IPv4

IPv4 addresses are 4 bytes long, written in decimal, i.e. 144.32.240.99 There are around 4.3 billion IPv4 addresses. The address is split into two sections: the network ID and the host ID, used for determining which network and which computer on the network an IP address is for.

Originally: 8 bit network ID, 24 bit host ID. (too few networks, too many unused hosts)

Class A: 0, 7bit netID, 24bit hostID
Class B: 1, 0, 14bit netID, 16bit hostID
Class C: 1, 1, 0, 21bit netID, 8bit hostID

Classful addressing allows allocation of different sizes of networks but not many organizations wanted class C addresses but would not fill class B addresses.

Classless Addressing instead uses a netmask to subdivide an IP address into a netID and hostID. Therefore both netID and hostID are variable.

IP address: 130.1.9.1 Netmask: 255.255.248.0, therefore first 21 bits is netID. Written as 130.1.9.1/24.

### 2.3.1 Subnets

Routers can further subdivide their IP address range to direct certain IP addressed to a different part of their LAN. Done in same way as classless addressing.

### 2.3.2 IPv4 Protocol

Connectionless, best effort Network Layer protocol.

IPv4 usually avoids using LLC for Ethernet, but does use LLC for Wi-Fi and other IEEE 802 networks.

**Version** 4 bits, 4 for IPv4.

**IP Header Length** 4 bits, measured in 32-bit words.

**Type of Service** 1 byte, unused except for congestion notification.

**Total length** 2 bytes, total length of packet in bytes.

**ID** 2 bytes, identify each fragment to be reassembled in order.

**Flags** 3 bits, 0, Dont Fragment/Do Fragment, More Fragments/No More Fragments

**Fragment Offset** 2 bytes - 3 bits, where the fragment should be stored within the entire datagram. Measured in 8 byte words.

**TTL** 1 byte, measured in hops.

**Protocol** 1 byte, which higher-layer protocol sent the IP packet.

**Header Checksum** 2 bytes, checksum of header only using Internet checksum.

**Source/Dest IP** 4 bytes each.

**Options** Variable length. Miscellaneous options for timestamps, probing and routing.

**Padding** Variable length.

## 2.4 UDP

Connectionless, best effort Transport layer protocol.

Header:

**Source/Dest Port** 2 bytes each.

**Length of Data** 2 bytes, measured in bytes.

**Checksum** 2 bytes, optional. Generated from Source/Dest IP, IP Protocol and UDP Length.

## 2.5 TCP

Connection-oriented, reliable Transport layer protocol.

Uses the idea of *streams* which are divided automatically into packets and sent over IP. Can also handle dynamic flow control by modifying window size based on max receiver size. Supports delayed ACK by including in header.

**Source/Dest Port** 2 bytes each.

**Sequence Number** 4 bytes. Sequence number of first byte in segment, measured in bytes. Used for Go-Back-N error control.

**Acknowledgement Number** 4 bytes. Next sequence number expecting to receive, measured in bytes.

**Data Offset** 4 bits, length of header in 32-bit words.

**Reserved** , 4 bits set to 0.

**Flags** 1 byte, set to various flags based on connection state.

**Receiver Window Size** 2 bytes, used for flow control.

**Checksum** 2 bytes, required. Similar to UDP checksum.

**Urgent Pointer** 2 bytes, sequence number of byte after urgent data, should be sent out of sequence to application layer.

**Options** Up to 40 bytes.

**Padding** Variable to pad to whole 32-bit word size.

TCP Connection Set-Up: SYN(Tx), SYN(Rx), ACK(Tx).

TCP Connection Tear-Down: FIN(Tx), ACK(Rx), FIN(Rx), ACK(Tx)

TCP Slow Start:

- Window size starts at 1
- If frames transmit with ACK double size until equal to receiver window size
- If frames fail, reduce window size.

### 2.5.1 Congestion Control

Try to control traffic to control flow on entire route instead of one link.

See Notes.....

## 2.6 DNS

DNS is used to convert human-readable names to IP addresses. Domain Names are hierarchal and made of different sub-domains.

Name servers are responsible for name resolution of full domain names. Each name server may defer to a higher or lower-level name server for a specific TLD. Name servers may cache the result of queries to prevent needing to pass a request around.

Primary name servers are the authority for their domain. Secondary name servers store copies of primary name servers to reduce traffic and provide a fallback if primary goes down.

DNS queries may be recursive (request resolution is handled by first DNS server called) or iterative (client must resend request to each DNS server in turn). DNS queries are called via TCP or UDP.

DNS stores information about each name in a database with additional information that may be relevant.

Resource Record:

**Domain Name**

**Class** IN for Internet.

**Type** A for IPv4, AAAA for IPv6, CNAME for domain name alias, MX for mailserver, etc.

**Data** Variable length dependant on type.

DNS Header:

**ID** 2 bytes, used for matching query to result.

**QR** 1 bit, 1 for response, 0 for query.

**Opcode** 4 bits, usually 0.

**Flags** 6 bits, returns details to do with authority of answer or recursion querying.

**Rcode** 5 bits, error code.

**Num of Queries** 2 bytes.

**Num of Answers** 2 bytes.

**Num of Name Servers** 2 bytes.

**Num of Addit. Records** 2 bytes.

## 2.7 IPv6

128 bit long addresses, written in sections of 2 byte hex. Classless addressing with netmasks. Host ID may contain MAC address to ignore the need for DHCP.

Local addresses start with FE80:0000:0000:0000 (usually written w/o zeros).

IPv6 addresses are hierarchical, with first few bits specifying country, therefore routers can only look at first few bits to get in correct country, then specify.

Routers may not fragment packets, instead dropping any that are too long.

IPv6 Header:

**Version** 4 bits, 6.

**Traffic Class** 1 byte, same as Type of Service.

**Flow Label** 20 bits, not used.

**Payload length** 2 bytes, length of data segment only in bytes.

**Next Header** 1 byte, type of next header (for additional headers) or data protocol type.

**Hop Limit** 1 byte, same as TTL.

**Source/Dest IP** 128 bits each.

Additional headers may be added after the main IPv6 header. This is done by checking the Next Header field for instructions. If the option is not known, consult the top 2 bits of the type:

**00** Skip to next header.

**01** Discoard packet silently.

**10** Discard packet and send ICMP packet to source router.

**11** Same as 11 for only unicast.

## 2.8 ICMP

Best effort, connectionless Network layer protocol used for relaying operational info and error messages between routers about IP.

Traceroute via ICMP ECHO:

- Send ECHO with IP TTL of 1.
- First router replies that IP packet exceeded TTL, so now know first router in chain.
- Increment TTL continuously and make note of new routers in chain.
- Eventually reach host and know entire route.

## 2.9 ARP

ARP is used on LANs for hosts to determine MAC address from known IP address. Makes use of the broadcast MAC address that allows every node on network to receive packet. Node with given MAC address responds with IP address, stored in table for future lookups.

## 2.10 DHCP

Used for dynamic IP allocation. Host sends broadcast with MAC FF-* and 255.255.255.255. Server replies to known MAC address with specified information.