Unix System Administration

IP Basics

Layers

- Complex problems can be solved using the common divide and conquer principle. In this case the internals of the Internet are divided into separate layers.
  - Makes it easier to understand
  - Developments in one layer need not require changes in another layer
  - Easy formation (and quick testing of conformation to) standards
- Two main models of layers are used:
  - OSI (Open Systems Interconnection)
  - TCP/IP

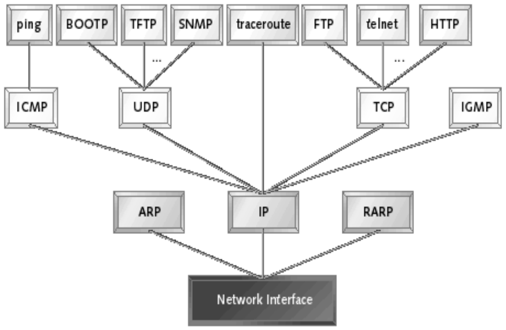
OSI

- Conceptual model composed of seven layers, developed by the International Organization for Standardization (ISO) in 1984.
  - Layer 7 – Application (servers and clients etc web browsers, httpd)
  - Layer 6 – Presentation (file formats e.g pdf, ASCII, jpeg etc)
  - Layer 5 – Session (conversation initialisation, termination,)
  - Layer 4 – Transport (inter host comm – error correction, QOS)
  - Layer 3 – Network (routing – path determination, IP[x] addresses etc)
  - Layer 2 – Data link (switching – media acces, MAC addresses etc)
  - Layer 1 – Physical (signalling – representation of binary digits)
- Acronym: All People Seem To Need Data Processing

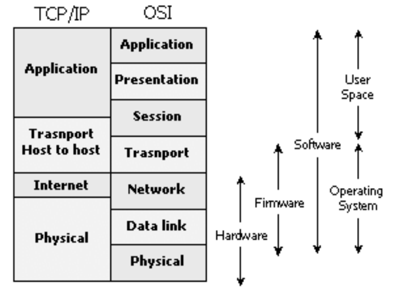
TCP/IP

- Generally, TCP/IP (Transmission Control Protocol/Internet Protocol) is described using three to five functional layers. We have chosen the common DoD reference model, which is also known as the Internet reference model.
  - Process/Application Layer consists of applications and processes that use the network.
  - Host-to-host transport layer provides end-to-end data delivery services.
  - Internetwork layer defines the datagram and handles the routing of data.
  - Network access layer consists of routines for accessing physical networks.

TCP/IP diagram



OSI and TCP/IP



IP datagram structure

<table>
<tr><td>Version</td><td>IHL</td><td>TOS</td><td colspan="2">Total Length</td></tr>
<tr><td colspan="3">Identification</td><td>Flags</td><td>Fragment Offset</td></tr>
<tr><td colspan="2">TTL</td><td>Protocol</td><td colspan="2">Header Checksum</td></tr>
<tr><td colspan="5">Source IP Address</td></tr>
<tr><td colspan="5">Destination IP Address</td></tr>
<tr><td colspan="4">Options</td><td>Padding</td></tr>
<tr><td colspan="5">Payload (TCP/UDP/ICMP etc.)</td></tr>
</table>

VERSION (4 bits)
The version field is set to the value '4' in decimal or '0100' in binary. The value indicates the version of IP (4 or 6, there is no version 5).

IHL (4 bits)
The Internet Header Length (IHL) describes how big the header is in 32-bit words. This allows the receiver to know exactly where the payload data begins.

TOS (8 bits)
Type of service allows the intermediate receiving stations (the routers) to have some notion of the quality of service desired.

TOTAL LENGTH (16 bits)
This is the length of the entire datagram in octets, including the header. This is why an IP datagram can be up to 65,535 bytes long, as that is the maximum value of this 16-bit field.

IDENTIFICATION (16 bits)
Sometimes, a device in the the middle of the network path cannot handle the datagram at the size it was originally transmitted, and must break it into fragments. If an intermediate system needs to break up the datagram, it uses this field to aid in identifying the fragments.

FLAGS (3 bits)
The flags field contains single-bit flags that indicate whether the datagram is a fragment, whether it is permitted to be fragmented, and whether the datagram is the last fragment, or there are more fragments. The first bit in this field is always zero.

FRAGMENT OFFSET (13 bits)
When a datagram is fragmented, it is necessary to reassemble the fragments in the correct order. The fragment offset numbers the fragments in such a way that they can be reassembled correctly.

TIME TO LIVE (8 bits)

This field determines how long a datagram will exist. At each hop along a network path, the datagram is opened and it's time to live field is decremented by one (or more than one in some cases). When the time to live field reaches zero, the datagram is said to have 'expired' and is discarded. This prevents congestion on the network that is created when a datagram cannot be forwarded to it's destination. Most applications set the time to live field to 30 or 32 by default.

PROTOCOL (8 bits)

This indicates what type of protocol is encapsulated within the IP datagram. e.g UDP, IGMP, ICMP

HEADER CHECKSUM (16 bits)

According to RFC 791, the header checksum formula is:*"the 16-bit ones compliment of the ones compliment sum of all 16-bit words in the header."*

The checksum allows IP to detect datagrams with corrupted headers and discard them. Since the time to live field changes at each hop, the checksum must be re-calculated at each hop. In some cases, this is replaced with a cyclic redundancy check algorithm.

SOURCE ADDRESS (32 bits)

This is the IP address of the sender of the IP datagram.

DESTINATION ADDRESS (32 bits)

This is the IP address of the intended receiver(s) of the datagram. If the host portion of this address is set to all 1's, the datagram is an 'all hosts' broadcast.

OPTIONS & PADDING (variable)

Various options can be included in the header by a particular vendor's implementation of IP. If options are included, the header must be padded with zeroes to fill in any unused octets so that the header is a multiple of 32 bits, and matches the count of bytes in the Internet Header Length (IHL) field.

Numbering Rules

- Private IP address ranges:
  - 10/8 (10.0.0.0 – 10.255.255.255)
  - 192.168/16 (192.168.0.0 – 192.168.255.255)
  - 172.16/12 (172.16.0.0 – 172.31.255.255)
- Public Address space available from AfriNIC
- Choose a small block from whatever range you have, and subnet your networks (to avoid problems with broadcasts)

Forwarding

- If a computer isn't on your subnet, packet's sent via a “gateway” connected to to networks.
- defaultrouter option in /etc/rc.conf sets the default gateway for this system.
- IP forwarding on a FreeBSD box turned on with the gateway_enable option in /etc/rc.conf otherwise the box will not forward packets from one interface to another.
- *paper routing exercise.*

Client – Server Arch

- Client makes requests, Server serves requests – e.g HTTP for transferring “websites”. This is the easiest way to provide services on demand and provides a means of sharing resources more effectively.
- Example: Mimicking the browser with telnet (client) talking to a web server (server)

```
telnet www.google.com 80
GET / HTTP/1.0
Host: www.google.com
<blank line>
```

Debugging

- ping
- traceroute
- tcpdump