The seven layers of networking

There is no doubt that modern CCTV gets involved in networking more and more each day. This is a new area to many in CCTV, to some it may appear complicated, to others a mystery, but we can not bury our heads in the sand and not learn about it. Many installers and system designers will call an IT expert in their team in order to clear and sort things out, but then not having even basic knowledge and not knowing what to discuss might prove extremely unproductive.

Luckily the networking theory and practice have been standardised and it is only a matter of learning the "tricks of the trade."

When we browse the Internet, a physical connection allows us to connect to the internet, through a modem (PSTN, ISDN) or through an Ethernet card (ADSL, Cable,...) in the case of a dedicated connection. Often the Ethernet network card is also called NIC for Network Interface Card.

A TCP/IP communication stack allows us to pass traffic and resolve web sites to IP addresses. Applications such as Mozilla (Netscape), Internet Explorer, Outlook and Eudora, allow us to see the web sites and receive our e-mail.

The modem or Ethernet function has 2 parts. The modem or Ethernet drivers provide the computer with a way to communicate with the hardware. The PPP (Point to Point Protocol) connection, also known as Dial-up Networking, allows your computer to access the modem. These two components provide the basis of getting a connection to the Internet.

The TCP/IP stack allows the computer to pass traffic across the link to the Internet in a meaningful way. That is, the TCP/IP stack allows your computer to speak the same “language” as the equipment at the other end of your connection. The TCP/IP stack also allows you to resolve friendly host names, such as www.cctvfocus.net, into an IP (Internet Protocol) address. Without the TCP/IP stack, we would be forced to go to each web site by its IP address instead of a name!

Finally, the applications allow us to interact with friendly software to interpret HTML code into web pages for us, interact with mail servers to exchange e-mail, connect to news servers to retrieve and post news articles, and exchange data with FTP servers to allow us to download files. Without these programs, the Internet would be much more difficult to navigate through.

The basics of networking revolves around understanding the so called *Seven layer OSI model*. Proposed by the ISO (International Standards Organization) OSI could be seen as ISO backwards, but it actually means *Open System Interconnection*.

Why was it created?

The principles that were applied to arrive at the seven layers are as follows:

- A layer should be created where a different level of abstraction is needed.
- Each layer should perform a well defined function.
- The function of each layer should be chosen in accordance with developing internationally standardized protocols.
- The layer boundaries should be chosen to minimize the information flow across the interfaces.
- The number of layers should be large enough that distinct functions need not be thrown together in the same layer out of necessity, and small enough that the architecture does not become unwieldy.

Having a way of categorizing each factor in an internet connection makes it **easier for us to do our jobs** as troubleshooters.

We all inherently understand that if the modem is not plugged in, you’re not going to be able to get your e-mail. The OSI model allows us

to follow that logic further: for example, if you can browse the web by IP but can't see websites by name, you know that the problem is not on the Network layer, but on the Transport layer.

The OSI Seven-Layer Model

The Open System Interconnection - OSI networking suite standard development began in the 1980s, by the European-dominated International Standards Organization (ISO).

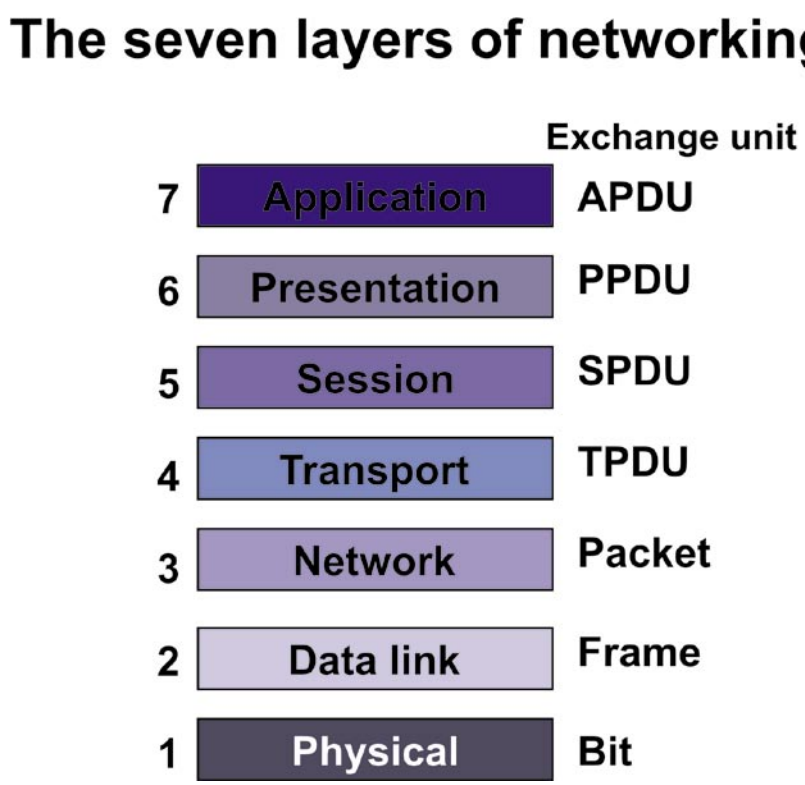
OSI has two major components: an abstract model of networking (the Basic Reference Model, or *seven-layer model*) and a set of concrete protocols. Parts of OSI have influenced Internet protocol development, but none more than the abstract model itself, documented in OSI 7498 and its various addenda. In this model, a networking system is divided into layers. Within each layer, one or more entities implement its functionality. Each entity interacts directly only with the layer immediately beneath it, and provides facilities for use by the layer above it. Protocols enable an entity in one host to interact with a corresponding entity at the same layer in a remote host.

The seven layers of the OSI Basic Reference Model are (from bottom to top):

1. Physical layer
2. Data link
3. Network
4. Transport
5. Session
6. Presentation
7. Application

1. The Physical layer

The Physical Layer describes the physical properties of the various communications media, as well as the electrical properties and interpretation of the exchanged signals. For example, this layer defines the size of Ethernet coaxial cable, the type of BNC connector used, the Cat5 twisted pair connections and the termination method. The physical layer is concerned with transmitting raw bits over a communication channel. The design issues have to do with making sure that when one side sends a 1 bit, it is received by the other side as a 1 bit, not as a 0 bit. Typical questions here are how many volts should be used to represent a 1 and how many for a 0, how many microseconds a bit lasts, whether transmission may proceed simultaneously in both directions,



how the initial connection is established and how it is torn down when both sides are finished, and how many pins the network connector has and what each pin is used for. The design issues here deal largely with mechanical, electrical and procedural interfaces and the physical transmission medium, which lies below the physical layer. Physical layer design can properly be considered to be within the domain of the electrical engineer.

2. The Data link

The Data Link Layer describes the logical organisation of data bits transmitted on a particular medium. This layer defines the framing, addressing and checksumming of Ethernet packets. The main task of the data link layer is to take a raw transmission facility and transform it into a line that appears free of transmission errors in the network layer. It accomplishes this task by having the sender break the input data up into data frames (typically a few hundred bytes), transmit the frames sequentially, and process the acknowledgment frames sent back by the receiver. Since the physical layer merely accepts and transmits a stream of bits without any regard to meaning of structure, it is up to the data link layer to create and recognize frame boundaries. This can be accomplished by attaching special bit patterns to the beginning and end of the frame. If there is a chance that these bit patterns might occur in the data, special care must be taken to avoid confusion.

The data link layer should provide error control between adjacent nodes.

Another issue that arises in the data link layer (and most of the higher layers as well) is how to keep a fast transmitter from "drowning" a slow receiver in data. Some traffic regulation mechanism must be employed in order to let the transmitter know how much buffer space the receiver has at the moment. Frequently, flow regulation and error handling are integrated, for convenience.

If the line can be used to transmit data in both directions, this introduces a new complication that the data link layer software must deal with. The problem is that the acknowledgment frames for A to B traffic compete for the use of the line with data frames for the B to A traffic. A clever solution in the form of piggybacking has been devised.

3. The Network layer

The Network Layer describes how a series of exchanges over various data links can deliver data between any two nodes in a network. This layer defines the addressing and routing structure of the Internet. The network layer is concerned with controlling the operation of the subnet. A key design issue is determining how packets are routed from source to destination. Routes could be based on static tables that are “wired into” the network and rarely changed. They could also be determined at the start of each conversation, for example a terminal session. Finally, they could be highly dynamic, being determined anew for each packet, to reflect the current network load.

If too many packets are present in the subnet at the same time, they will get in each other’s way, forming bottlenecks. The control of such congestion also belongs to the network layer.

Since the operators of the subnet may well expect remuneration for their efforts, there is often some accounting function built into the network layer. At the very least, the software must count how many packets or characters or bits are sent by each customer, to produce billing information. When a packet crosses a national border, with different rates on each side, the accounting can become complicated.

When a packet has to travel from one network to another to get to its destination, many problems can arise. The addressing used by the second network may be different from the first one. The second one may not accept the packet at all because it is too large. The protocols may differ, and so on. It is up to the network layer to overcome all these

problems to allow heterogeneous networks to be interconnected.

In broadcast networks, the routing problem is simple, so the network layer is often thin or even nonexistent.

4. The Transport layer

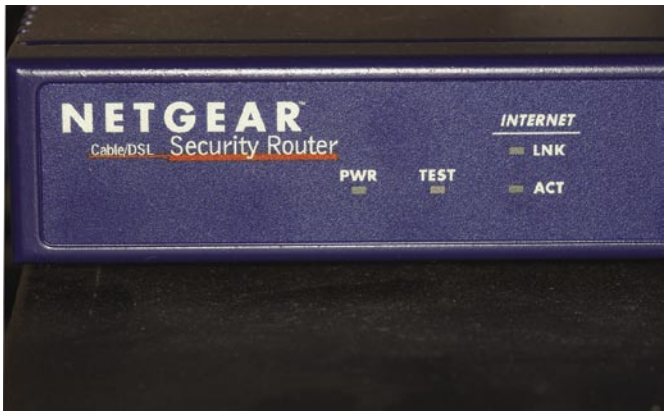
The Transport Layer describes the quality and nature of the data delivery. This layer defines if and how retransmissions will be used to ensure data delivery. The basic function of the transport layer is to accept data from the session layer, split it up into smaller units if need be, pass these to the network layer and ensure that the pieces all arrive correctly at the other end. Furthermore, all this must be done efficiently and in a way that isolates the session layer from the inevitable changes in the hardware technology.

Under normal conditions, the transport layer creates a distinct network connection for each transport connection required by the session layer. If the transport connection requires a high throughput, however, the transport layer might create multiple network connections, dividing the data among the network connections to improve throughput. On the other hand, if creating or maintaining a network connection is expensive, the transport layer might multiplex several transport connections onto the same network connection to reduce the cost. In all cases, the transport layer is required to make the multiplexing transparent to the session layer.

The transport layer also determines what type of service to provide to the session layer, and ultimately, the users of the network. The most popular type of transport connection is an error-free point-to-point channel that delivers messages in the order in which they were sent. However, other possible kinds of transport, service and transport isolated messages with no guarantee about the order of delivery and broadcasting of messages to multiple destinations. The type of service is determined when the connection is established.

The transport layer is a true source-to-destination or end-to-end layer. In other words, a program on the source machine carries on a conversation with a similar program on the destination machine, using the message headers and control messages.

Many hosts are multi-programmed, which implies that multiple connections will be entering and leaving each host. There needs to be some way to tell which message belongs to



which connection. The transport header is one place this information could be put.

In addition to multiplexing several message streams onto one channel, the transport layer must take care of establishing and deleting connections across the network. This requires some kind of naming mechanism, so that process on one machine has a way of describing with whom it wishes to converse. There must also be a mechanism to regulate the flow of information, so that a fast host cannot overrun a slow one. Flow control between hosts is distinct from flow control between switches, although similar principles apply to both.

5. The Session layer

The Session Layer describes the organisation of data sequences larger than the packets handled by lower layers. This layer describes how request and reply packets are paired in a remote procedure call. The session layer allows users on different machines to establish sessions between them. A session allows ordinary data transport, as does the transport layer, but it also provides some enhanced services useful in a some applications. A session might be used to allow a user to log into a remote time-sharing system or to transfer a file between two machines.

One of the services of the session layer is to manage dialogue control. Sessions can allow traffic to go in both directions at the same time, or in only one direction at a time. If traffic can only go one way at a time, the session layer can help keep

track of whose turn it is.

A related session service is token management. For some protocols, it is essential that both sides do not attempt the same operation at the same time. To manage these activities, the session layer provides tokens that can be exchanged. Only the side holding the token may perform the critical operation.

Another session service is synchronisation. Consider the problems that might occur when trying to do a two-hour file transfer between two machines on a network with a 1 hour mean time between crashes. After each transfer was aborted, the whole transfer would have to start over again, and would probably fail again with the next network crash. To eliminate this problem, the session layer provides a way to insert checkpoints into the data stream, so that after a crash, only the data after the last checkpoint has to be repeated.

6. The Presentation layer

The Presentation Layer describes the syntax of data being transferred. This layer describes how floating point numbers can be exchanged between hosts with different math formats. The presentation layer performs certain functions that are requested sufficiently often to warrant finding a general solution for them, rather than letting each user solve the problems. In particular, unlike all the lower layers, which are just interested in moving bits reliably from here to there, the presentation layer is concerned with the syntax and semantics of the information transmitted.

A typical example of a presentation service is encoding data in a standard, agreed-upon way. Most user programs do not exchange random binary bit strings. They exchange things such as people's names, dates, amounts of money, and invoices. These items are represented as character strings, integers, floating point numbers and data structures composed of several simpler items. Different computers have different codes for representing character strings, integers and so on. In order to make it possible for computers with different representation to communicate, the data structures to be exchanged can be defined in an abstract way, along with a standard encoding to be used "on the wire". The job of managing these abstract data structures and converting from the representation used inside the computer to the network standard representation is handled by the presentation layer.

The presentation layer is also concerned with other aspects of information representation. For example, data compression can be used here to reduce the number of bits that have to be transmitted and cryptography is frequently required for privacy and authentication.

7. The Application layer

The Application Layer describes how real work actually gets done. Ex: this layer would implement file system operations. The application layer contains a variety of protocols that are commonly needed. For example, there are hundreds of incompatible terminal types in the world. Consider the plight of a full screen editor that is supposed to work over a network with many different terminal types, each with different screen layouts, escape sequences for inserting and deleting text, moving the cursor, etc.

One way to solve this problem is to define an abstract network virtual terminal for which editors and other programs can be written to deal with. To handle each terminal type, a piece of software must be written to map the functions of the network virtual terminal onto the real terminal. For example, when the editor moves the virtual terminal's cursor to the upper left-hand corner of the screen, this software must issue the proper command sequence to the real terminal to get its cursor there too. All the virtual terminal software is in the application layer.

Another application layer function is file transfer. Different file systems have different file naming conventions, different ways of representing text lines, and so on. Transferring a file between two different systems requires handling these and other incompatibilities. This work, too, belongs to the application layer, as do electronic mail, remote job entry, directory lookup, and various other general-purpose and special-purpose facilities.

The original Internet protocol specifications defined a four-level model, and protocols designed around it (like TCP) have difficulty fitting neatly into the seven-layer model.

Most newer designs however use the seven-layer model. [•]