STRIKE PUBLICATIONS

Defence today

DEFENCE CAPABILITIES MAGAZINE

November/December 2006
$7.95

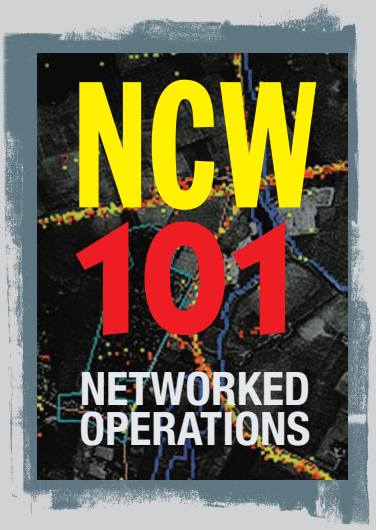
ABRAMS fires up

Tiger ARH under fire

Malaysian Su-30s surpass F/A-18s

New trends in coastal defence

Print Post PP424022/00254
ISSN 14470446



TCP/IP Networking Protocols

Dr Carlo Kopp

NCW 101 part 9

The Internet protocol suite, colloquially known as 'TCP/IP protocols' is without doubt the global standard for connecting computing systems. This protocol suite has displaced all other contenders over the past decade and now forms the backbone of global commercial, academic, government and military networks.

When the idea of the Internet was conceived by researchers and DARPA in the US more than three decades ago, little could they have anticipated its growth or indeed its popularity. This technology is now as much a household item as it is the core of global networking.

The prevalence of Internet protocols is a recent development, as proprietary protocol suites such as IBM SNA and Digital DECNet/DDCMP dominated most installations in the developed world as late as the 1980s. The Internet protocol suite based on TCP/IP family protocols did not emerge until the early 1980s, and at that time was primarily confined to academic and research computers running variants of the AT&T and BSD Unix operating system. In its infancy it faced strong competition from the OSI protocol suite, ostensibly an international standard, and during the 1980s and 1990s adopted by US Federal government agencies as the Government OSI protocol suite, or GOSIP. GOSIP compliance seemed to be a mandatory part of Australian Federal government agency tender requirements through this period despite the reality that only a handful of software applications existed that could actually run over the GOSIP protocols. Happily, the non-viable GOSIP has since vanished from the wish-list of acquisition bureaucrats globally, but it still represents an excellent case study of the bureaucratic passion for magical yet non-viable technology solutions.

The Internet protocol suite continued to steadily penetrate the global market, until 1994 when the World Wide Web (WWW/W3) started gathering serious momentum. From that point onward the fate of GOSIP and the proprietary protocols was sealed in the mass market. The W3 grew exponentially, riding on top of the Internet protocol suite.

What distinguished the Internet protocol suite from other contenders was that it was built from the outset for 'internetworking', or connecting geographically disparate networks together, and it was designed to run on top of entrirely arbitrary datalink channels. These could be voiceband modems running over phone lines or voice radios, wireless or infrared datalinks or networks, or cabled datalinks such as Ethernet, Token Ring, ADSL or others. In short, the Internet protocol suite was built for global reach, and it had the flexibility to run on top of almost any conceivable cabled or wireless channel. Its third key attribute was that it was 'open' in the sense that it was an industry standard which anybody could use without royalties, and it was extensible, with a standards process designed to make it easy to add additional functions, modes or protocols to the suite.

As a result, the Internet protocol suite was able to evolve much faster than proprietary or proprietary pseudo-standard protocol suites, and thus adapt to the historically mind boggling growth in network size and performance during the 1990s. While the Internet protocol suite has its origins in US DoD DARPA funding, launched during the 1960s (http://www.livinginternet.com/i/ii.htm), the technology has only seen large scale military use over the past decade, and even now only a handful of operationally deployed systems actually have connectivity using these protocols. This is changing and it is a fair assessment that most key US platforms and systems will have such connectivity by the end of the coming decade, at varying data rates depending on what datalink channels are available.

While the Internet protocol suite is designed to run over almost any conceivable datalink channel, this capability is not automatic, and typically the manner in which the protocols are interfaced requires a unique definition, and usually a standard. The PPP protocol is preferred for many channels, including commodity ADSL.

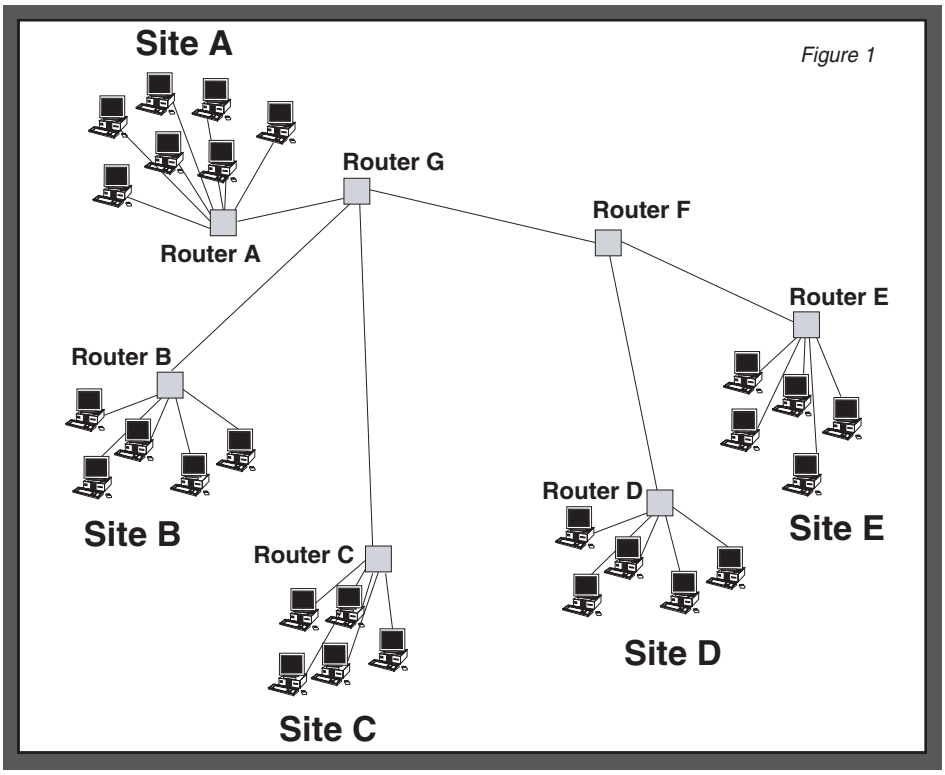


Figure 1

How the Internet Protocol Suite Works

To appreciate how the 'TCP/IP' protocols function, we need to look at the network from two perspectives. The first is 'topological', exploring how the network is connected and how traffic flows, the second is in terms of the layers of protocols being used.

The functional and theoretical basis of the Internet is a technique called 'packet routing', devised by then MIT PhD student Leonard Kleinrock during the 1960s. Packet routing is based on the idea of taking a communication between two computers, chopping it into small pieces termed 'packets', encapsulating these packets with addressing and other information, and then having these packets flow through a network of devices termed 'routers'. Routers were built as computers that had multiple communications interfaces, the sole purpose of which was to accept incoming packets, decode the addressing information, then send them on their way using the appropriate communications interface. Each router contained a map of the network topology - the specific manner in which routers were interconnected in the network - and using this map and the addressing information, it could determine exactly which communications interface it needed to use to get a packet travelling in the right direction to get to its intended destination.

The then new idea of a packet network was powerful and led to the definition of the 'catenet' model of networks, the basis of today's Internet. During the 1960s the US DoD funded this research, via DARPA, leading to the initial ARPANET, and eventually, today's Internet.

The Internet and proprietary networking schemes - most of which are long extinct - use a defacto hierarchical topological model for interconnecting individual nodes in the network, refer Figure 1. In this model, multiple computers at any given site are connected to a router, for instance via a Local Area Network. If any of these computers intends to communicate with a peer at another site, the packets it sends must traverse multiple 'hops' between routers until they reach their intended destination.

An example might be a computer at Site B communicating with a computer at Site D. The packets travelling in either direction must traverse hops between Routers B, G, F and D. If we imagine a physical network, then the hop between B and G might be a high speed serial link over a copper cable, the hop between F and D might be a high speed optical fibre link, and the hop between G and F a microwave link with dish antennas on towers.

This trivial example is hierarchical and highly structured. The uppermost level in the hierarchy is occupied by Routers G and F, the lowermost level by routers A, B, C, D and E. All connections forming the network topology are known beforehand. The Internet follows this basic model but is immensely more complex, with millions of computers and routers connected to form its topology.

This model was devised for fixed infrastructure networks, using mostly copper and optical fibre connections throughout. Wireless networks using the 802.11 protocols, now increasingly a ubiquitous feature in many portable computers and devices, emulate this model.

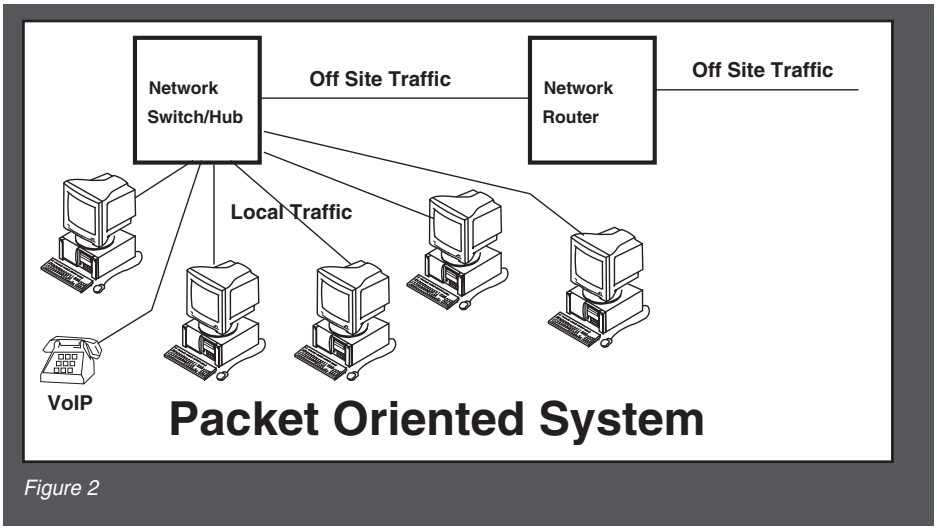


Figure 2

In such wireless networks, a radio-frequency link in the 900 MHz, 2.45 GHz or 5.8 GHz band is used to connect the computer to a local network, and usually the lowermost router in the local hierarchy. In effect, such wireless networks replace the last cabled connection between the computer and the network. Wireless it may well be, but otherwise it is little more than an extension of the fixed infrastructure network.

Connecting routers to allow traffic to flow is not enough to make it happen. A key question is how to tell every router in the network where it should send packets in order to reach a specific destination. For all intents and purposes this is the problem of addressing.

In the Internet scheme, we are familiar with network names in a text form, such as editor@defencenews.com.au (email) or www.defencenews.com.au (WWW). If we wish to send an email or access a website, our computer must first query the network to get a network address, produced by translating the name into an Internet Protocol or IP address, in this instance 202.148.146.201. Next, our computer must open a stream connection to the machine addressed as 202.148.146.201. As the first packet is sent, each router along the way looks up this address, to determine which direction to send the packet in. Eventually, the packet arrives at 202.148.146.201, which responds accordingly. This is a gross simplification, but important to explain how the network functions.

Reality is more complicated, as every query to discover a new name to address mapping must be directed to a 'Domain Name Server', which is a computer running DNS software which maintains a directory of name to address mappings. As no single computer could realistically cope with such a number of queries, the Internet uses a very large number of redundant DNS servers, all organised in a virtual hierarchy. If a DNS server does not know a specific mapping, it queries the server above it in this hierarchy, and so on, until a server is found which knows the answer.

Once the DNS server provides the computer with the required IP address, it can initiate a connection via its nearest router. That router has to understand enough of the topology of the network to know where to direct the connection. Again, there is further hidden complexity in this 'route discovery' process. Routers maintain what are termed 'routing tables' which contain mappings between addresses and specific interfaces. Large routers may indeed have dozens of interfaces to other routers, and this information must be continuously maintained for the whole network.

To distribute and manage this information, routers have to communicate with one another using specific protocols. The most widely used of the older protocols is Routing Information Protocol (RIP), with Open Shortest Path First (OSPF) being preferred in newer systems. When a router receives a packet, it uses the cached information gathered with the protocol to calculate what it believes is the best route to the destination. Suffice to say, there is further complexity in this mechanism that is too extensive to discuss here.

To summarise, every computer connected to the network must have a network address, as we will soon see, this is an IP address, and routers use these addresses to determine where to route incoming packets. Supporting protocols for route discovery and maintenance are required for routers to understand the topology of the network. In addition, text based names must be mapped into addresses, and this is performed using the DNS.

At the protocol level, there is inevitably further complexity, but it is necessary for network function. Datalink protocols (discussed in NCW 101 Pt 2) provide a means of carrying data between individual devices, such as machines on a local network, or routers connecting two sites. These protocols are however limited to local addresses, and have no mechanisms to permit routing between sites. Moreover, such protocols usually lack mechanisms for managing traffic flow rates, which is critical in large networks with links of often dissimilar speeds.

To address the range of needs in a large network requires multiple protocols, and this led to the adoption of the 'layered' model for protocols, using a 'stack' of protocols. A simple example is illustrative.

Assume we have two computers that wish to communicate a block of data across a large network. Refer Figure 2. The first problem to be solved is that of both using a mutually compatible application program to understand the data, so the protocol stack must accommodate this with an abstraction called the 'application layer'.

Having solved that problem, the next which arises is that of data format compatibility, ie ensuring that the data is delivered in a fashion which is readable by the other machine in the connection. This abstraction is called the 'presentation layer' and good examples are conversion between IBM EBCDIC and industry ASCII formats, or the MIME encoding used with email attachments.

The next problem to solve is managing the state of the connection, ie the equivalent of lifting a phone off the hook, dialling and putting it down once the conversation is over. Management of session state is thus handled by yet another abstraction, termed the 'session layer'.

At this point the message is wrapped with application formatting information, is coded into some format using the presentation layer, and has attached control information to manage the state of the connection. What is now needed is a protocol layer that can transport the encapsulated message across a network, reliably or unreliably. This is performed by the 'transport layer', yet another abstract label. Transport protocols assume a connected path of routers exists between the two machines, therefore another layer is required to handle routing of traffic and accommodate the idiosyncrasies of the datalinks being used. This function is performed by the 'network layer', which handles routing but also fragments or aggregates network layer packets as required by the underlying datalink protocol. These may have unique oddities such as packet size limits.

Datalink protocols are represented by a 'datalink layer' and the physical channel, such as a cable or radio link, by a 'physical layer'.

There are seven layers of abstraction that form the OSI model for a network, one which is very widely used in practice:

Layer 1: Physical Layer
Layer 2: Data Link Layer
Layer 3: Network Layer
Layer 4: Transport Layer
Layer 5: Session Layer
Layer 6: Presentation Layer
Layer 7: Application Layer

Any message being sent between two computers must flow down the OSI 'stack' at the transmitting end, and flow up the stack at the receiving end. With every layer it crosses on the way down the stack, it is encapsulated or wrapped up with additional information required by that layer, on the way up the stack this information is interpreted, used and stripped away. A good analogy is an onion with seven layers of skin, the core of which is the message. Refer Figure 3.

Internet Protocol

The Internet Protocol or IP sits one layer above the Datalink Layer protocols, which are used to effect point-to-point transmission of packets. IP is primarily used to move packets across the network, hopping between nodes. IP is a simple protocol and does not provide error correction, reliability in delivery if packets are trashed or lost in transmission. What it does provide are key capabilities of unique addressing the source computer and the destination computer, and performing fragmentation and reassembly of messages if the message size is larger than the maximum packet size permitted by a specific datalink protocol running between any two routers along the link.

IP addressing is effected in IPv4 (Version 4) using a 32-bit (four-byte) 'Internet Address', permitting up to 4.3 billion unique addresses in a network. Since the 1990s it was clear that growth of the Internet would exhaust this pool of addresses, although there is no current consensus on exactly when this will happen, so IPv6 (Version 6) was defined to replace IPv4, using a larger 128-bit (sixteen-byte) address to provide decades of headroom in address space occupancy.

User Datagram Protocol

The User Datagram Protocol or UDP sits one layer above the Internet Protocol, and is used as an unreliable or 'best effort' transport protocol to support numerous different applications. Examples of applications that run over UDP include streaming video, streaming audio, Voice over IP telephony and the aforementioned DNS mechanism.

The UDP header is very simple, and in its basic form does little more than identify the application protocol it is carrying via its 'port number'. Ports are exit and entry points defined in a host computer's network interface, which are typically specific to a particular application protocol. While IP addresses are essential to permit packets to flow between computers, the applications that are generating and receiving specific packets must be known, and port numbering provides for this. Commonly used port numbers are 20 and 21 for File Transfer Protocol; 25 for Simple Mail Transfer Protocol; 22 for encrypted Secure Shell; 53 for DNS; 80 for Hypertext Transfer Protocol or http; 109 and 110 for POP email retrieval protocol; 6000, 6001 for X11 graphic windowing; 5060 for SIP Voice over IP, etc. Many applications will dynamically allocate ports on demand, presenting interesting security problems.

While UDP is often ignored in popular discussion of the Internet protocol suite, it is an essential and important protocol.

Transmission Control Protocol

Transmission Control Protocol or TCP is the most commonly discussed transport protocol used on the Internet. Unlike its simpler sibling UDP, TCP is a reliable protocol that attempts to transmit packets without loss and deliver them in the proper order to the receiving application. TCP also includes a mechanism to manage the flow of packets across routers in the network and avoid congestion as required. Summarising TCP's capabilities:

1. Reliable delivery – lost packets are detected and resent.
2. In order delivery – out of order packets are delivered in order.
3. Duplicate rejection – duplicated packets are discarded.
4. Flow and congestion control – limitations in router throughput are accommodated.

The reliability and quality of transmission provided by TCP comes at a cost, both in terms of protocol complexity, but also in problematic behaviour, then running across error or packet loss prone channels. In the latter instance, the time it takes a packet to transit the network can vary widely if retransmission is required frequently, making TCP ill-suited for applications where time critical traffic must be carried, such as Voice over IP or streaming multimedia.

The Future

Predicting the long term future of the Internet protocol suite is fraught with difficulty, since the technology is evolving so rapidly. While the core protocols including IP, TCP and UDP remain stable, and are apt to see principal changes over the coming decade in the adoption of IPv6 addressing, the suite itself is continuously growing as more application protocols are added over time, and others adapted to changing needs. Some idea of the scope of this evolution can be gained by browsing the Internet Engineering Task Force (IETF) website at http://www.ietf.org.

The biggest near term change we can expect to see is the introduction en masse of Voice over IP installations, replacing legacy circuit switch analogue and digital telephony.

While much of the discussion and debate on NCW necessarily focuses on wireless datalink channels, which are the fundamental bottleneck in military networks, increasing levels of support for IP over such datalinks will progressively alter the form of military networking over time, as increasing numbers of networked applications emerge. The notion of an IP connection to a bomb's or cruise missile's internal computer in flight is an emerging reality.

Further Reading: A vast number of good websites and texbooks exist which detail the inner workings of the Internet protocol suite. Recommendations include Wikipedia pages on these topics.

Figure 3

OSI Model			
	Data unit	Layer	Function
Host layers	Data	Application	Network process to application
		Presentation	Data representation and encryption
		Session	Interhost communication
	Segments	Transport	End-to-end connections and reliability
Media layers	Packets	Network	Path determination and logical addressing (IP)
	Frames	Data link	Physical addressing (MAC & LLC)
	Bits	Physical	Media, signal and binary transmission

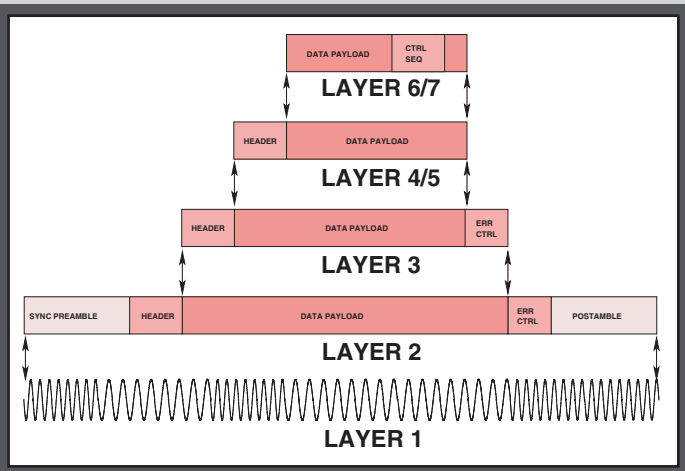


Figure 4
Mostly encapsulation involves attaching a header to a message, although sometimes it may also involve attaching a trailer, or recoding and encrypting a message. Headers and trailers are typicallly fixed sized blocks of addressing and control data, formatted in a fashion specific to a specific protocol. Given the three protocols of most interest here, the Internet Protocol (Layer 3), User Datagram Protocol (Layer 4) and Transmission Control Protocol (Layer 4), only headers are involved. (Image via Wikipedia)