XoT White Paper

December 2008

The purpose of this white paper is to explain how XoT works, how it is intended to be utilized, and what role it can play in migrations from X.25 networks to TCP/IP based networks. We show how Gcom's Protocol Appliance (GPA) can be utilized in XoT networks in novel ways.

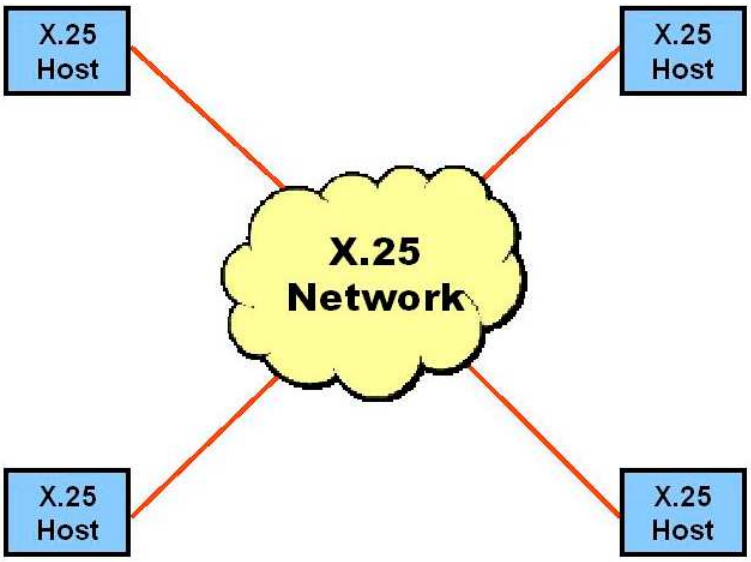
What is XoT

XoT is an acronym for X.25 over TCP. It is described in RFC 1613 and was invented by Cisco Systems. But before getting into XoT let us take a historical view of the evolution of X.25 networks so that we can see the context for XoT.

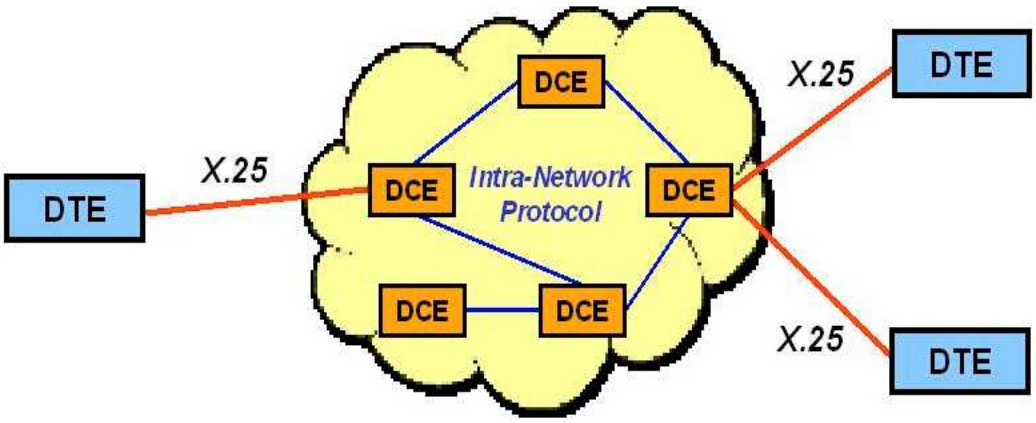
X.25 in the Beginning

In the simplest view, an X.25 network is a constellation of data communication elements that allow X.25-capable host computers to communicate with one another. X.25 networks allow for many host computers that are geographically dispersed.

The original impetus for such networks came from the rise, in the 1970s, of multinational corporations which had a need for data services among their many dispersed places of business worldwide. This is why the X.25 standard was created by a United Nations body, the CCITT, as an international standard, rather than as a standard within a single country or, like SNA, within a single computer systems manufacturer.



Now, let's look at these X.25 networks in more detail and introduce some X.25 terminology.



In X.25 parlance, the host computers are referred to as "packet mode DTEs". This is not to be confused with the use of the term DTE in describing the RS232 electrical interface. A packet mode DTE means an X.25-capable computer.

The equipment to which the DTEs are connected, at the edge of the X.25 network, is referred to as a packet mode DCE. Once we know that we always mean "packet mode" we can shorten these to just DTE and DCE.

The link between the DTE and DCE is, in the model of the standard, a synchronous data line. Or, if you will, a leased line from the phone company. Note that the label X.25 is applied to this access line and not to the entire network, as we did in the first drawing. This is because the CCITT specified X.25 as in interfacing protocol, not as an end-to-end protocol.

Inside the network each network provider has its own internal protocols that it uses to connect the DCEs. Not shown are intermediate network nodes that are never connected to DTEs. These intermediate nodes are only concerned with the internal networking protocols. The internal protocols do not necessarily bear any resemblance to X.25. All that matters is that the DCEs use the X.25 procedures to operate the access line to the DTEs.

Different Strokes

There tended to be two different approaches to the internal network protocols: virtual circuit and datagram.

In the virtual circuit approach a Call Request packet from one DTE would cause the network to find a path across the network to the destination DTE and its corresponding DCE. The network would remember this path and packets on that X.25 logical channel (from either DTE) would traverse this same path. Typically a "circuit number" would be assigned to each path for reference.

In the datagram approach what was remembered was the network address of the remote DCE. Each packet was routed independently of all others and the paths that any two packets took on the same logical channel could be completely different.

Each technique had its advantages and its drawbacks.

The advantage of the virtual circuit method is that all a given switching node needs to know about a packet when it arrives is the previously discovered next interface port to steer the packet to. So routing a packet is a matter of looking up in a table by virtual circuit number in which the table entry points to the outbound interface. Queue the packet for that interface and you are done.

The advantage of the datagram method is that it can use redundant pathways to the remote DCE automatically, so load balancing across the whole network becomes automatic. It also can route around line and node failures since there is no commitment to either ahead of time.

The drawback to the virtual circuit method is that when a line or a node does fail then there has to be re-routing of the virtual circuit. Since both ends may perceive the failure at the same time there are possible races involved in which the two ends seek to re-route at the same time. These have to be resolved.

The drawback to the datagram method is that there is no reliability in getting the packet from one DCE to the destination DCE. Thus, some DCE to DCE protocol needs to be imposed on top of the datagram delivery service to ensure that the packets arrive intact, and are put back in order.

Back in the 1970s and 1980s the computer technology used in the switching nodes was slow enough that the amount of extra computation necessary for the datagram method could be significant. It makes a difference whether it takes 500 instructions or 10,000 instructions to route a packet if instructions come at the rate of a micro-second or so apiece.

Likewise, back then trunk lines could be 56K or 64K. At such slow speeds it was of importance to keep the header size down in the internal networking protocols. Datagrams, because of having to carry full addresses, tended to be longer headers than simple circuit number-based headers.

On paper the datagram method had much to recommend it, but the case could not be closed because of practical computing concerns.

Then Came IP

Later network trunk lines became faster and the switching computers became even faster yet. The amount of computation applied to each packet to be switched was no longer such an overriding concern.

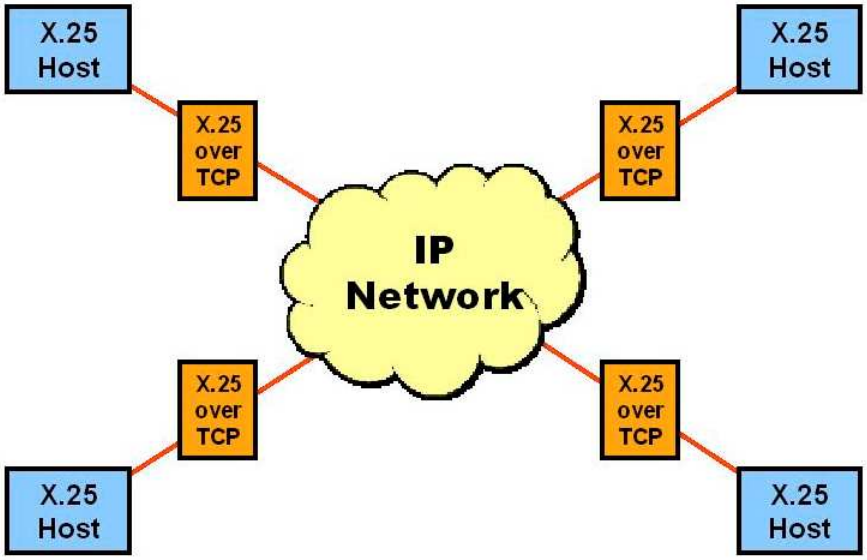
With the invention of the TCP and IP protocols for the ArpaNet, for LANs and subsequently for the Internet, there were now standards for intermediate node protocol (IP) and for end-to-end protocol (TCP). Now everyone wanted to take advantage of the bandwidth optimizing features of datagram protocols.

Network providers began building IP networks and offering attractive services to customers based on their ability to make much better use of their trunk line bandwidth. Network managers were motivated to take advantage of these services.

But I Still Have X.25 Hosts

The problem was that many companies had X.25 host computers and were still using X.25 networks. Replacing all those host computers would offset the savings of using the new network technology, and then some. So how to retain the X.25 hosts and still reap the benefits of the new IP networks?

What was called for was something like the following.



If you can place a box in between the X.25 host and the IP network, and if that box can communicate using X.25 to the host and TCP/IP to the network, and if a group of such boxes can all communicate with each other via the IP network, then the problem is solved.

This is what Cisco came up with in RFC 1613: X.25 over TCP -- XoT. In fact, TCP and IP are the perfect combination to re-architect X.25 networks. Recall the previous discussion of the datagram method of routing packets through an X.25 network. Well, IP is a datagram protocol and IP networks have all the advantages of that technique. And TCP is an end-to-end reliable protocol. So, in effect, the boxes labeled “X.25 to TCP” play the role of the DCEs in our original drawing of X.25 networks.

Problem solved.

OK, So How Does It Work?

Let’s go into a little detail about how XoT works. We will look at the protocol headers of X.25 and of XoT and see how they relate to one another.

This is what an X.25 packet looks like when viewed from the vantage point of the access line.

The LAPB Hdr is two bytes, the Pkt Hdr is 3 bytes, each Flag is one byte and the CRC is two bytes. So there is not much header overhead in an X.25 packet.

The purpose of the LAPB layer is to transport the X.25 packets reliably across the access line which, if you recall, is assumed to be a telco line and thus subject to errors.

In order to send X.25 packets over TCP, we want to use TCP as this reliable protocol instead of LAPB. So we are looking for something like the following.

The only problem is that TCP is a byte stream protocol. It will not preserve the message boundaries of the original X.25 packets. So what is needed is just two more bytes that give the length of the X.25 packet so that the packet boundaries can be re-established on the receiving end.

This is the essence of the XoT protocol. Basically you take the 3-byte X.25 packet header, put a byte count in front of that and send the resulting message over a TCP connection. The receiving end uses the byte count to re-establish the packet boundary, strips off the byte count, adds a LAPB header and sends it over the access line to the X.25 Host.

In XoT there is a new TCP connection for every X.25 logical channel. This differs from X.25 in which a single LAPB link multiplexes connections for many X.25 logical channels.

There is a special procedure that must be performed when the X.25 Host sends an X.25 Call Request packet. The XoT implementation must associate the X.121 DTE address in the Call Request packet with the IP address of the remote XoT to which it is to be sent. Once that TCP connection is established, however, XoT simply sends the Call Request packet as-is over the TCP connection, so the remote X.25 Host sees the original Call Request packet and can interpret facilities and user data. But XoT did not have to be concerned with the fine points of X.25 facilities. This simplifies the XoT implementation.

In order to most easily implement XoT it is best to have an X.25 access line that is dedicated to the use of XoT. You run a LAPB protocol stack on that line. This allows XoT to remove the Byte Cnt header from XoT packets and simply transmit the resulting packet over the LAPB line as-is (with a little bit of logical channel mapping).

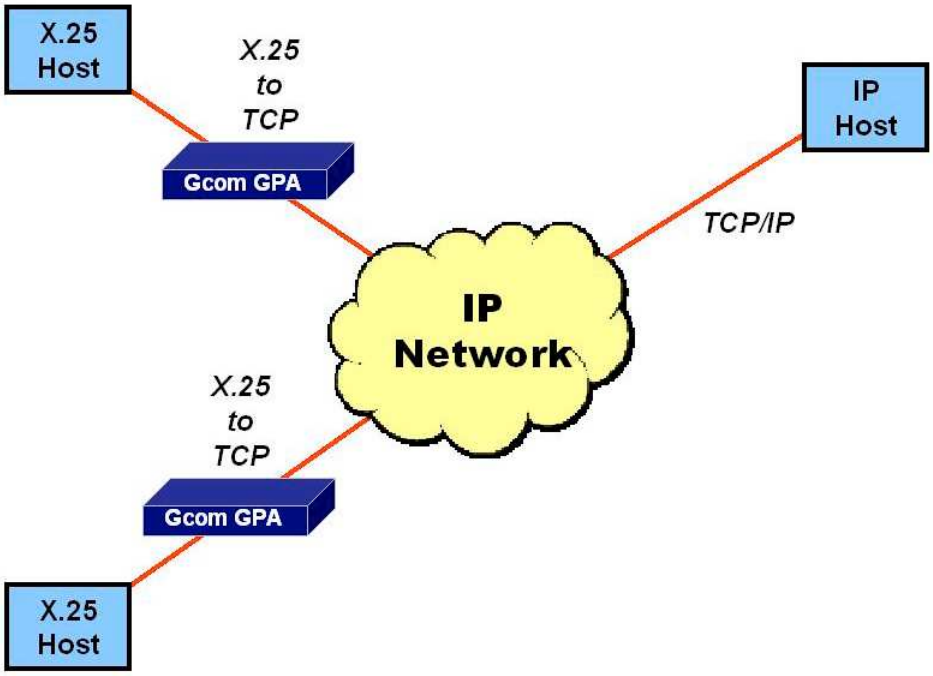
This is essentially how the Gcom XoT implementation works. You simply dedicate one or more ports on a GPA to XoT X.25 traffic and XoT routes the packets from those ports, each running a LAPB stack, to TCP connections over Ethernet.

So What's The Problem?

As long as your networking requirements call for X.25 on both ends of the network, XoT works just fine. But if one side of the network is X.25 and the other is TCP/IP, XoT can no longer function properly. This is because XoT needs to send received XoT packets over a LAPB link. It is not able itself to perform all the functions of X.25 so as to be able to remove the X.25 packet level header, thus reducing the packet to pure data.

Situations such as this arise when remote sites are still using legacy X.25 equipment but at the host end it is desired to eliminate the X.25 links to the host. The motivation for this may be that the X.25 links could very well be the only remaining ports in use on an expensive Front End Processor. Except for those ports the host could be pure TCP/IP and the FEP could be decommissioned. This is increasingly important as more mainframe manufacturers withdraw support for their FEPs.

There are two methods to get around this problem, both supported by Gcom.



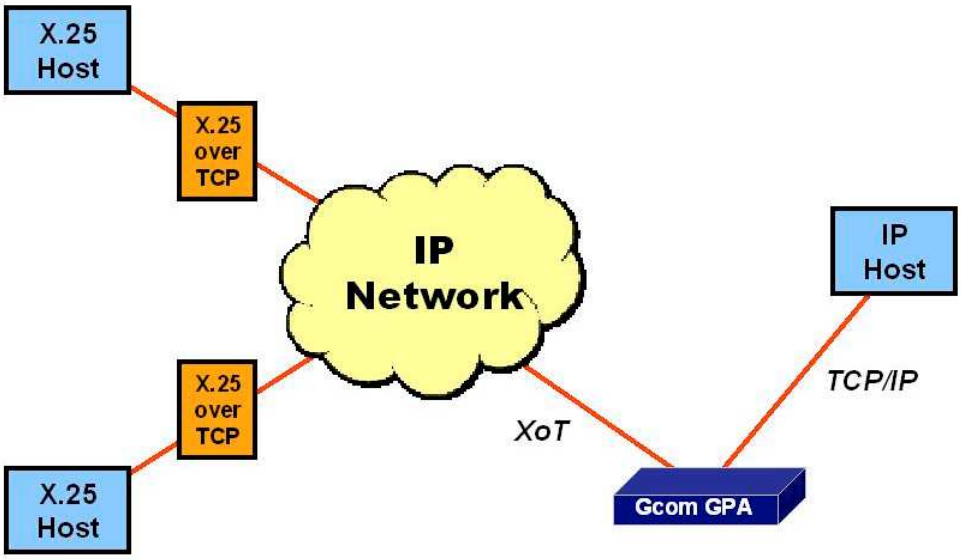
In this method a Gcom GPA terminates the X.25 and sends just the data portion of the packet across a TCP connection, with a byte count header added.

The packets coming from the X.25 host look like this.

And the packets sent over the IP network look like this.

Notice that the GPA does not send the X.25 packet header as XoT does. Thus the IP Host need only utilize the Byte Cnt to establish the message boundary and then process the Data field of the message. Gcom's GPA supports over a dozen different forms of the Byte Cnt header, including omitting it altogether for self-identifying data such as ISO 8583 messages.

The second method involves continuing to send XoT over the IP network, but converting to encapsulated data on the host end. This scheme looks like the following:



The packets coming into the GPA from the IP network look like this.

And the packets going to the IP Host look like this.

The Byte Cnt in the packets to the host can be chosen from among Gcom's list of encapsulation headers.

The GPA accomplishes this by feeding the X.25 from its XoT module into an X.25 stack within the GPA itself. The GPA's X.25 stack removes the X.25 packet header and performs all the necessary procedures of X.25. The resulting Data then flows through the GPA just as if it came from an external X.25 link. This includes being routed to Gcom's Protocol Converter Daemon,

the module responsible for managing the TCP connections to the IP Host and adding the encapsulation header.

Conclusion

When migrating X.25 networks to TCP/IP one needs to address the protocol issues on both the host end and the remote end of the network. Often it is preferable to eliminate XoT from the network in order to broaden the alternatives of choosing vendors or application software packages at the host end. This is especially true when multiple host sites are involved. Several cases can be distinguished:

1. Entire network is still X.25

The idea is to eliminate the X.25 network and replace it with a TCP/IP network. If nothing changes either at the host end or the remote end, then a simple application of XoT at both ends will suffice and be transparent.

However, if it is desirable to change the host end to a TCP/IP based application and then gradually change the remotes from X.25 to TCP/IP, then one of the Gcom solutions becomes appropriate. If multiple hosts are involved it could be advantageous to place Gcom GPAs at the remote sites to terminate the X.25 and forward payload data in simple encapsulated TCP packets. This is an especially effective solution if the remotes need to interact with different hosts, perhaps provided by different service providers. In such cases it is impractical to convince the different providers to install GPAs at the host sites to translate XoT to encapsulated TCP.

2. Network is XoT based, migration of remotes not required

In this case the remote sites are already sending XoT packets over an IP backbone. If it is desired to change the host application to TCP, then a GPA at the host end makes sense to terminate the XoT and forward payload data to the host as simple encapsulated TCP packets.

If the multiple host requirement enters the picture then, just as in example 1, the best course of action is to place a GPA at each remote site and eliminate the XoT from the network at that point. This configuration has the maximum flexibility in terms of utilizing multiple servers from multiple vendors.

It may also be the case that the XoT routers are sufficiently old that they are no longer supported by the router vendor. In this case they must be replaced with newer, supported equipment. In such a case a GPA that eliminates the XoT from the network makes the most sense.

3. Network is XoT based, migration of remotes required

In this case it is desired not only to change the host application to TCP but also, over time, to replace the X.25 remote equipment with native TCP/IP equivalents. The best approach in this case is to place GPAs at the remote sites so that the entire network operates as a TCP/IP network. Individual remote sites can then be upgraded with no impact on the network.