

Bridging

Bridging is a method of path selection that contrasts with routing.

In a bridged network, no correspondence is required between addresses and paths. Put another way, addresses don't imply anything about where hosts are physically attached to the network. Any address can appear at any location. In contrast, routing requires more thoughtful address assignment, corresponding to physical placement.

Bridging relies heavily on broadcasting. Since a packet may contain no information other than the destination address, and that implies nothing about the path that should be used, the only option may be to send the packet everywhere! This is one of bridging's most severe limitations, since this is a very inefficient method of data delivery, and can trigger *broadcast storms*. In networks with low speed links, this can introduce crippling overhead.

IP, designed as a wide-area networking protocol, is rarely bridged because of the large networks it typically interconnects. The broadcast overhead of bridging would be prohibitive on such networks. However, the link layer protocols IP functions over, particularly Ethernet and Token Ring, are often bridged. Due to the pseudo-random fashion in which Ethernet and Token Ring addresses are assigned, bridging is usually the only option for switching among multiple networks at this level.

Bridging is most commonly used to separate high-traffic areas on a LAN. It is not very useful for disperse traffic patterns. Expect it to work best on networks with multiple servers, each with a distinct clientele that seldom communicate with any servers but their "home".

Two types of bridging exists, corresponding to the distinction outlined earlier. *Transparent bridging* is used in Ethernet environments and relies on switching nodes. Token Ring networks use *source-route bridging* (SRB), in which end systems actively participate by finding paths to destinations, then including this path in data packets.

Transparent bridging

Transparent bridging, the type used in Ethernet and documented in IEEE 802.1, is based on the concept of a *spanning tree*. This is a tree of Ethernet links and bridges, spanning the entire bridged network. The tree originates at a *root bridge*, which is determined by election, based either on Ethernet addresses or engineer-defined preference. The tree expands outward from there. Any bridge interfaces that would cause loops to form are shut down. If several interfaces could be deactivated, the one farthest from the root is chosen. This process continues until the entire network has been transversed, and every bridge interface is either assigned a role in the tree, or deactivated.

Since the topology is now loop-free, we can broadcast across the entire network without too much worry, and any Ethernet broadcasts are flooded in this manner. All other

packets are flood throughout the network, like broadcasts, until more definite information is determined about their destination. Each bridge finds such information by monitoring source addresses of packets, and matching them with the interfaces each was received on. This tells each bridge which of its interfaces leads to the source host. The bridge recalls this when it needs to bridge a packet sent to that address. Over time, the bridges build complete tables for forwarding packets along the tree without extraneous transmissions.

There are several disadvantages to transparent bridging. First, the spanning tree protocol must be fairly conservative about activating new links, or loops can develop. Also, all the forwarding tables must be cleared every time the spanning tree reconfigures, which triggers a *broadcast storm* as the tables are reconstructed. This limits the usefulness of transparent bridging in environments with fluid topologies. Redundant links can sit unused, unless careful attention is given to root bridge selection. In such a network (with loops), some bridges will always sit idle anyway. Finally, like all bridging schemes, the unnecessary broadcasting can affect overall performance. Its use is not recommended in conjunction with low-speed serial links.

On the pro side, transparent bridging gives the engineer a powerful tool to effectively isolate high-traffic areas such as local workgroups. It does this without any host reconfiguration or interaction, and without changes to packet format. It has no addressing requirements, and can provide a "quick fix" to certain network performance problems. As usual, careful analysis is needed by the network engineer, with particular attention given to bridge placement.

Again, note that for IP purposes the entire spanning tree is regarded as a single link. All bridging decisions are based on the 48-bit Ethernet address.

Source-route bridging (SRB)

Source-route bridging (SRB) is popular in Token Ring environments, and is documented in IEEE 802.5. Unlike transparent bridging, SRB puts most of the smarts in the hosts and uses fairly simple bridges. SRB bridges recognize a routing information field (RIF) in packet headers, essentially a list of bridges a packet should transverse to reach its destination. Each bridge/interface pair is represented by a *Route Designator (RD)*, the two-byte number used in the RIF. An *All Rings Broadcast (ARB)* is forwarded through every path in the network. Bridges add their RDs to the end of an ARB's RIF field, and use this information to prevent loops (by never crossing the same RD twice). When the ARB arrives at the destination (and several copies may arrive), the RIF contains an RD path through the bridges, from source to destination. Flipping the RIF's *Direction Bit (D)* turns the RIF into a path from destination to source. See RFC 1042 for the format of the RIF field and a discussion of SRB's use to transport IP packets.

Source-route bridging has its problems. It is even more broadcast-intensive than transparent bridging, since each host must broadcast to find paths, as opposed to each bridge having to broadcast. It requires support in host software for managing RIF fields. To take advantage of a redundant network, a host must remember multiple RIF paths for

each remote host it communicates with, and have some method of retiring paths that appear to be failing. Since few SRB host implementations do this, SRB networks are notorious for requiring workstation reboots after a bridge failure.

On the other hand, if you want to bridge a Token Ring network, SRB is just about your only choice. Like transparent bridging, it does allow the savvy engineer to quickly improve network performance in situations where high-traffic areas can be segmented behind bridges.