

1

Hardware Design

Hardware design may, at first sight, seem to be something over which the network architect or designer has no control. It would appear, therefore, that it plays very little part in the design of a scalable Internet Provider (IP) network. Despite this, an understanding of the various architectures employed by the hardware vendors can influence the choice of hardware. Thus, the choice of hardware will inevitably affect your network designs.

There are only a finite number of architectures on which hardware designs are based. Most vendors of core Internet routers today use various Application Specific Integrated Circuits (ASIC) and/or network processors designed specifically for the purpose of performing IP and MPLS lookups, applying access control/firewall functionality, enforcing policy and applying Quality of Service (QoS). However, there are significant differences in the various implementations of architectures within the plethora of available access routers and switches. Some still rely on a software-based system with a more general-purpose processor, which provides the flexibility to add services at the expense of forwarding performance. This contrasts with ASICs, which are (as the name suggests) designed for a single function, or group of functions. The first generation of ASIC-based devices provided very high-speed forwarding at the cost of flexibility and services. In addition, this improved performance is gained at the cost of higher development cost and longer time to market. The introduction of network processor-based systems has combined the flexibility of a general-purpose processor without sacrificing any of the packet forwarding capabilities of an ASIC. In addition, the non-specific nature of the network processor can lead to lower costs than those associated with ASICs.

1.1 SEPARATION OF ROUTING AND FORWARDING FUNCTIONALITY

One of the major steps forward in ensuring scalability in network performance over the past few years has been the move towards the separation of routing and forwarding functionality. Put simply, the process of calculating the forwarding table is run on different hardware from that responsible for actually implementing the packet forwarding.

By separating these functions onto different hardware modules it is possible to continue forwarding packets at line rate even in the presence of severe disruptions to the network topology. Designs based on general-purpose processors have to use processor cycles to make both the routing decisions and the forwarding decisions. Therefore, if the network topology is dramatically and repeatedly updated, the processor with a fixed number of cycles is required to take cycles from the forwarding tasks to carry out the routing tasks or vice versa.

1.2 BUILDING BLOCKS

A router based on the principles of separation of routing and forwarding functionality is based on a few simple building blocks, (see Figure 1.1).

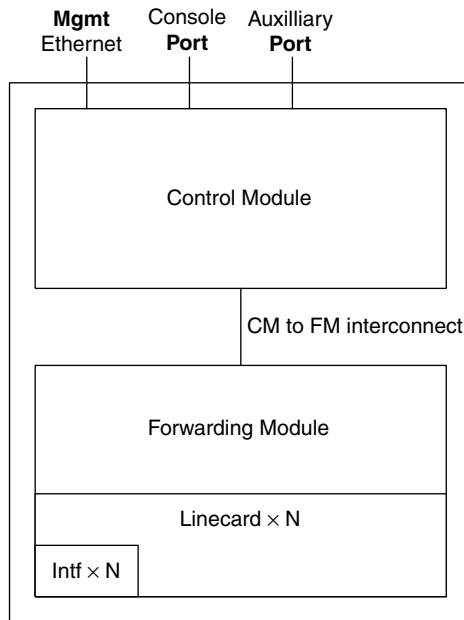


Figure 1.1 The logical architecture of an IP router (separate forwarding and routing planes)

1.2.1 CONTROL MODULE

The control module is responsible for running the routing protocols, creating a routing table from the sum of the routing information received and generating a forwarding table from the routing table. This includes the enforcement of policy relative to the routing information being imported and exported. In addition to the routing function, this module provides Command Line Interface (CLI), management and logging functionality.

1.2.2 FORWARDING MODULE

The forwarding module is responsible for the receipt of packets, decoding the Layer 1, Layer 2 and Layer 3 information and sending that packet to the appropriate output interface (or interfaces in the case of multicast) where it is re-encoded with Layer 3, Layer 2 and Layer 1 information before transmission onto the wire. The forwarding module is also the ideal location for implementing access control/firewall filters and class/quality of service. ASIC and Network Processor-based devices tend to implement these features in the forwarding module. Traditional, software-based routers always had to send every packet to the CPU if any of these features were enabled. This was one of the main reasons why software-based routers were unable to implement complex services while maintaining high packet throughput.

1.2.3 NON-STOP FORWARDING

Non-stop forwarding is the generic term for the ability of a router to carry on forwarding packets even though the device responsible for the calculation of the routing and forwarding tables is temporarily inoperable. The specific reason for the inoperability of the routing device is really of no importance at this stage of the discussion. Non-stop forwarding fundamentally relies on the separation of routing and forwarding functionality within the router. In addition, each routing protocol requires a number of modifications to the protocol operation in order to prevent the clearance of routes associated with the failed protocol from the forwarding table on the forwarding modules, both on the local router and on the neighbours of the local router, while the routing process or module recovers.

1.2.4 STATEFUL FAILOVER

This takes non-stop forwarding concept one step further. In a switch or router with redundant control modules it is possible to duplicate the state associated with routing protocols from the master routing module to the slave routing module. The slave control module is held in a state in which it can ‘instantly’ take over the operation of a failed master control module. Generally, the slave can take over operations within a matter of seconds.

There are, as usual, strongly held opinions about whether this is a good idea or not. Some vendors feel that it is foolhardy to copy all the state from the active control module to the slave. The main concern is that if a particular software bug or combination of state causes the master control module to reset and that state has been copied faithfully onto the slave, then, in all likelihood, the same problem will arise on the slave and cause it to reset. This would clearly be something of a disaster, given that the whole point of mirroring the state onto the slave control module is to prevent the total failure of the device.

On the other side of the argument, this mechanism provides the potential for the fastest failover. By maintaining an exact copy of the state from the master control module on the slave control module, it is possible to begin forwarding packets immediately without having to rebuild the routing information bases (RIB). This mechanism imposes no extra requirements upon the routing protocols to enable non-stop forwarding.

1.3 TO FLOW OR NOT TO FLOW?

This might sound like a rather bizarre question. The obvious answer is ‘of course we want the traffic to flow’. But this question does not actually relate to the flow of traffic but to the internal architecture of the network device.

Many Layer 2 switches work on flow-based mechanisms. A flow is a single stream of traffic associated with a single ‘conversation’ between two network devices. These can be uniquely identified by the source and destination address and source and destination TCP or UDP port. When the first packet for a new flow arrives, the packet is sent to the CPU and the switch creates a new entry in a flow table, which identifies the flow characteristics along with the ingress and egress ports. Once the flow entry has been created, then all subsequent traffic matching that flow is switched from the ingress to the egress port without touching the CPU, thus dramatically improving the performance. In an environment in which there are a constrained number of long-lived flows, these devices provide extremely good performance. However, if there are a large number of short-lived flows, performance becomes constrained by the ability of the CPU to create new entries in the flow table and by the amount of memory available to store the flow table. If there are too many new flows, either new ones will not be created or existing ones will have to be purged from the flow table to make way for the new ones. As soon as another packet for the purged flow arrives, it requires the set-up of a new flow. Thus, the problem is exacerbated.

As the line between Layer 2 devices and Layer 3 devices became blurred, many flow-based switches acquired Layer 3 (and Layer 4 through 7) functionality. In an enterprise environment, this is not a significant issue because flows are generally fewer in number and longer-lived. However, in a service provider there are almost invariably vast numbers of comparatively short-lived flows, particularly at major aggregated egress points. This is the worst possible situation for flow-based devices.

I have experienced instances of service providers installing flow-based devices at an Internet Exchange Point (IXP) and, very quickly, replacing those routers with routers of traditional design because of the intolerable performance of those devices in that role. You

may be asking what the attraction of such devices was given their clear disadvantages within a service provider network. The simple answer often is cost. Switch-based routers (a.k.a. Layer 3 switches) are invariably cheaper than their traditional counterparts, sometimes by a factor of three or four. They also generally have hybrid functionality so they can be used in both Layer 2 and Layer 3 roles. These advantages in terms of price and flexibility are often sufficient to convince hard-pressed service providers to try to get more for their money.

To be fair to the vendors of flow-based routers, many have enhanced their devices with much more powerful CPUs and much more memory for the flow tables. This helps to mitigate some of the problems but does not solve the inherent issues. The problem is just masked until the number of flows or rate of creation of flows grows sufficiently to overwhelm the new hardware.

One mechanism for overcoming this issue is tunnelling, since the packets are forwarded based on the outer headers, which relate only to the end points at the edge of the service provider's network. One widely used implementation of this feature is MPLS (discussed more fully later in the book). Because tunnels or Label Switched Paths (LSPs) are built between the edge routers in the networks and all traffic between those points is carried within the LSP, it aggregates to a single 'flow'.

Even with the enhanced devices, you should think long and hard before you use a flow-based router for a service provider network. Think particularly about the environment in which the router will be used and be sure that the performance constraints of the flow-based router will not come back and bite you.

1.4 HARDWARE REDUNDANCY, SINGLE CHASSIS OR MULTI CHASSIS

The choice of mechanism used to provide redundancy for each function within a single network node (a.k.a. hub) usually arouses one of two generally strongly held opinions. One theory is that it is best to create a single network element in which all the major service-impacting elements are replicated. This works on the premise that you generally provide $N + 1$ ($N \geq 1$) processing, switching, forwarding/interface, power and cooling modules so that if one module fails, the router can continue to function without any degradation in service. As stated above, failure of the control modules can have a brief impact on the normal routing behaviour of the device but forwarding should operate undisturbed (assuming certain preconditions are met). Failure of the switching fabric, the internal 'infrastructure' over which packets pass when travelling between linecards, should not have any impact on forwarding with the exception of the loss of any packets actually traversing the failed fabric at the instant it fails. Failure of forwarding/interface modules is more difficult to do without any impact because there usually has to be some element of external support for this (i.e. the far end has to also support the same feature—APS, dual PHY, etc.). This approach also means that there is generally a requirement for less power and rack space. There is also a reduction in the demands placed on your routing

protocols. Fewer links and fewer nodes mean less state and fewer changes in topology. Finally, the reduced administrative and support costs of operating a single device rather than several devices can be significant.

An alternative theory is that no matter how clever the design (or how much money you spend), it is not possible to produce a single unit with sufficient resilience in all the aspects (particularly forwarding) to adequately protect customer traffic. Therefore, it is better to build a resilient function by provisioning devices in groups, so that the complete failure of a single device may lead to reduced performance but never total loss of connectivity. However, more chassis inevitably mean a greater demand for power, more real estate, more instability in the network topology and greater support costs.

As with most things, designing a network is full of compromises and what usually happens is that service providers end up implementing a combination of the two theories. Traditionally, core routers, in which the forwarding function is paramount, are often provisioned in pairs with each taking a share of the links from other core sites and each having an equal capacity into the hub. The loss of one core router, therefore, reduces the aggregate capacity into the hub but does not isolate the hub from the rest of the network. The negative side of this model is that it significantly increases complexity. Due to the need to interconnect these boxes and to connect all edge boxes to both core routers, it inherently increases the number of interfaces required to service a single hub, thus increasing capital costs. Also, the added maintenance associated with more interfaces means greater operational costs. As the support for non-stop forwarding improves, it becomes more realistic to have single core routers in the core, thus dramatically reducing costs. As always, this is a big positive. The main issue is one of trust. If there is only one core router connecting a PoP to the rest of the network, the risk associated with the loss of that device is related to the number of edge routers, and therefore customers, connecting to the core via that single router. The question you have to ask yourself is whether you trust a router enough to be able to rely upon it to meet your SLAs. For instance, if one software upgrade requires a 6-minute reload, that would invalidate a 99.999% uptime SLA.

With edge routers, particularly access routers to which customers often have a single connection, it is not usually possible to usefully deploy them in pairs. Therefore, more of the single chassis redundancy features have been applied more widely to edge routers. If customers have only a single connection to a single router, it is essential that each router has resilient power, cooling, switching and control modules and, as far as is possible, resilient forwarding modules. The same risks to an SLA apply as above. However, the number of individual customers which a single failure can affect is generally significantly lower for edge routers because core routers typically aggregate several edge routers and their associated customers, thus, the impact is much greater with core routers.