

# SoftRouter: An Open Extensible Platform For Tomorrow's Internet Services

T. V. Lakshman K. Sabnani, T. Woo  
Bell Labs, Alcatel-Lucent  
600 Mountain Avenue  
Murray Hill, NJ, 07974.

## 1 Introduction

The rapid growth in the number and diversity of end-systems with limited processing, and the diversity of access technologies used for network access will result in a proliferation of intelligent, application-specific devices that will need to be deployed at the network edges. This proliferation will be necessitated by the many application-specific adaptations that will have to be applied to the packet stream by the network in order to accommodate the diversity of access networks and devices, and to avoid placing too much processing burden on the end-systems with limited battery or processing capability.

With cell-phones far outnumbering PCs, the number of end-systems with very restricted processing capability is exploding. Also, the diversity of devices that act as end-systems running specialized applications is also exploding – examples are set-top boxes connected to IP networks, sensors for wireless telemetry applications, etc. These diverse end-systems are also connected to the network over many different access networks with widely varying characteristics and with access bandwidths ranging from several tens of Mbps per customer to a few hundred kilobits.

These systems have capabilities that are inherently different from that of end-systems that have typically been connected to the Internet. An example is wireless clients that may be dormant for long periods of time, be intermittently

connected, and may be constrained by low battery capacities to limit the amount of processing done by them. Since a defining principle of the Internet architecture has been the end-to-end principle where much of the intelligence is concentrated at the end-systems with the network primarily providing connectivity, these new limited-capability clients require the network edge to provide the corresponding application-specific functions. Exceptions to the original end-to-end architecture vision have been increasing with the deployment of more and more mid-boxes deployed at the network edges to perform functions that are best done in the network. Examples are firewalls, NAT boxes and more recently application accelerators. The move toward the increased use of application-aware packet processing at the edges is likely to rapidly accelerate in the near future, resulting in a network where end-systems with limited processing are inter-connected through an application-aware network edge to a core network that provides the connectivity by traditional packet forwarding.

With current network architectures, this multitude of mid-box functions will each be done separately. A stream of packets for a particular application (as for example a VoIP stream) may traverse a network firewall, a session border controller, a transcoding device, and a router all within the same facility. Similarly, streams from other applications may traverse a sequence

of other application-specific devices. A video stream may traverse a transcoder, a client-specific ad-insertion device, a streaming cache and a firewall, all in sequence at a network edge.

The use of a multitude of separate application-specific devices has several disadvantages, both for the network provider and for end-users. From a network provider's perspective, since all the application-specific devices are in the forwarding path, each of these devices must meet the same redundancy and reliability requirements that routers typically need to meet. Furthermore, many basic packet processing functions need to be performed repeatedly in each of these devices. This includes traffic management to enforce diff-serv requirements, packet classification for identifying streams, and repeated session-state reconstruction for functions such as ad-insertion and firewalling. This increases the complexity of each of the individual network elements, wastes many ports for interconnect of these devices thus reducing port density, replicates the same functions unnecessarily in these difference devices, increases management complexity due to the many devices that need to be managed, and considerably increases the complexity of fault isolation and diagnostics for any one application.

This spectrum of application-aware processing that needs to be done at the edges is best done by a single open-platform that not only performs basic IP packet-processing but also enables packet streams to be processed through a programmable sequence of application-specific functions. The open platform enables the incorporation of best-in-class software for each of the application specific functions. The use of a single open-platform avoids many of the disadvantages due to the use of multiple network elements. Common functions such as traffic management, packet classification and session re-construction need to be done only once. Protocols for reliability (such as VRRP) need to be incorporated in only one network element. The need to manage

multiple devices and wastage of ports for inter-element interconnects is eliminated. Also, fault-diagnosis and isolation is easier since the state and diagnostics information need not be coordinated amongst multiple network elements.

In summary, for the evolving network-edge infrastructure packet forwarding is not the only prime function. The network-edge devices will also increasingly need to provide embedded application-aware packet functions alongside the traditional packet transport functions. This requires flexible network elements at the edges that can forward packets as well as perform programmable packet-payload and header operations that are customizable for each packet-stream.

These new network elements can be used for application-aware virtualization using a combination of deep-packet inspection and programmable data paths. For this, the packets are classified into forwarding equivalence classes using not only the L2/L3 header information but also information deeper in the packet header. Examples are URLs in the http header, strings in the packet-payload indicative of specific applications, etc. Once packets are classified in an application-aware manner, packets from each forwarding equivalence class can be processed in a manner specific to application needs. Policies can be specified to define the processing rules for each application. For example, some VoIP streams can be directed through a monitoring subsystem, video to certain users can be transcoded, and traffic to certain URLs can be redirected. Once the application-specific processing has been completed, these packets can be transported across a virtual topology in the core network using pre-established MPLS tunnels to provide isolation in a manner similar to current VPNs. Incorporation of these functions results in packet-streams from each application traversing their own virtual network with application-specific processing at the edges.

We illustrate this using the packet-flow

through an example networking platform, as shown in Figure 1, used for application-aware processing at the network edge. The platform consists of a standard chassis, such as an ATCA-based platform. Each slot in the chassis can be populated by packet-processing line-cards, service cards, control-plane processors or feature servers. The mix of line cards, service-cards, and feature-server cards is configurable by the service provider. Packets arriving at an ingress line card are classified into sub-streams by the line-card for application-aware processing. A sample classification would be the extraction of all packets that are destined to a particular destination prefix. This identified sub-stream can then be routed to service cards that performs a specific function such as running the sub-stream through an intrusion-detection application. On exiting this service card, the sub-stream may be further split – one sub-stream may be identified as a VoIP sub-stream and may be routed to a transcoding service card. The rest of the packets may go through a different sequence of service cards before exiting through an appropriate egress interface. We view the processing of each sub-stream as being accomplished by a series of service cards that perform application-aware processing on the packet stream. Note that in this example, the specific functions performed are application-specific by design and the functions performed may change when application needs change. Also, the packet-flows inside this platform change as new applications whose needs are very different from the current applications get deployed. Basic packet forwarding functions such as packet classification, address look-ups, traffic management, etc. constitute only one group amongst the many application-specific functions that get performed on the packet stream. The control and feature-server cards have open-interfaces that can be used for building custom control functions such as application-specific routing.

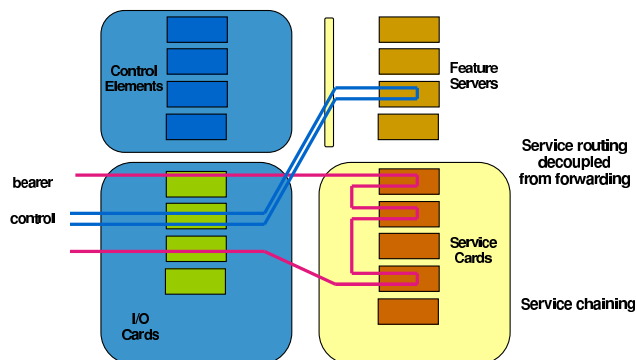


Figure 1: Example: Packet and Control Flow Through Network-Edge Device

## 2 Application-Aware Network Element Requirements

In the following, we describe a number of desirable characteristics for an application-aware network element.

### **Integration of services with forwarding:**

Application-aware networking extends packet-forwarding by adding customized packet processing functions to the normal forwarding functions. These network-edge platforms need the ability to classify packet streams based on information that is beyond the IP header and the ability to route packets through a sequence of service cards. Also it is necessary to transfer context information between service cards in a standardized manner to avoid performing the same functions (such as session state reconstruction) in multiple service cards.

### **Reprogrammable data and control path:**

A key characteristic needed is the ability to adapt packet processing to accommodate rapidly changing needs. In particular, changes to the data and control path functions can occur at any time due to changes in application-behavior or the need to adapt packet streams to new client or access network requirements. To meet this challenge, the networking platform must have reprogrammable data and control paths.

### **Separation of control and forwarding:**

With the extension of functions due to the use of multiple service cards, a particular chassis may have only a few line-cards with pure packet-forwarding capability with the rest being application-specific service cards. Dedicated control-cards on each chassis may not be necessary and for flexibility it is desirable to be able to centralize or distribute control as needed. This leads to the need for separation of control and forwarding with the controllers having the capability to control multiple forwarding cards using a standard protocol without any restrictions to controlling only components within the same chassis.

**Open and extensible system:** Besides being fixed in functions, traditional system designs are not based on standardized interfaces between the major components and new feature-development is constrained. For application-aware devices, the lack of open interfaces constrains rapid adaptation to changing application needs and prevents easy incorporation of best-in-class components for different features and services. It is necessary to architect the system to be open and extensible, and be in a position to incorporate new features in an efficient way. This is especially true for an IP services platform. A suitable networking platform will be extensible and allow efficient integration of external components.

### 3 SoftRouter as An Extensible Platform

The SoftRouter incorporates the characteristics described in the last section. Specifically, the SoftRouter platform integrates support of IP services with routing and switching, thus allowing easy integration into existing network configurations for access to packet streams that require IP service processing. Also, the SoftRouter separates the forwarding and control functions in an open system that redistributes the functionality amongst the Softrouter's forwarding elements,

service cards, control plane servers and feature servers. Refer [1] for a detailed description of this aspect of the SoftRouter, and refer [2, 3] for other approaches to redistributing of network functions. SoftRouters support for IP services is uniquely enabled via three system concepts:

1. Reprogrammable service cards: Packet applications are loaded on demand onto reprogrammable service cards. Reprogrammability allows system functions to evolve as requirements change.
2. Service chaining: This permits multiple packet-processing functions to be chained together in a customized manner to perform application-specific functions on a packet stream. The chaining is configurable on demand through software control. Service chains implement application bundles.
3. A comprehensive service management framework: Each application chain is monitored for diagnostics and resource usage. The system supports a rich set of high availability mechanisms to ensure automatic recovery with minimal flow disruption upon system faults.

SoftRouter provides a flexible way to configure how packet flows can traverse and get serviced by a sequence of service cards. Using a mix of off-the-shelf and custom-built cards, SoftRouter benefits from the ability to integrate a large number of commercial service cards that range from fixed functions (e.g., ASIC-based) to highly reprogrammable ones (e.g., CPU-based). Its flexibility through open interfaces permit it to ride the technology curve which is especially critical in areas such as wireless networks where evolving standards mean packet functions are changing rapidly and underlying technology improvement can often provide a very big performance jump.

These factors allow SoftRouter to be re-configured and extended for multiple applications without significant redevelopment effort,

making the SoftRouter approach well-suited for application-aware networking.

## References

- [1] T. V. Lakshman, T. Nandagopal, Ram Ramjee, K. Sabnani, and Thomas Woo The SoftRouter Architecture *Proceedings of HOTNETS 2004*, San Diego, CA.
- [2] Jennifer Rexford, Albert Greenberg, Gisli Hjalmysson, David A. Maltz, Andy Myers, Geoffrey Xie, Jibin Zhan, and Hui Zhang Network-Wide Decision Making: Toward A Wafer-Thin Control Plane *Proceedings of HOTNETS 2004*, San Diego, CA.
- [3] Nick Feamster, Hari Balakrishnan, Jennifer Rexford, Aman Sheikh, Jacobus van der Merwe The Case for Separating Routing from Routers *Proceedings of FDNA 2004*.