White Paper

Juniper Networks Solutions for Network Accounting

Chuck Semeria
Marketing Engineer

Hannes Gredler
Professional Services Engineer

Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, CA 94089 USA
408 745 2000 or 888 JUNIPER
www.juniper.net

Part Number : 200010-001 07/01

Contents

List of Figures

List of Tables

Executive Summary

In order to maximize their return on investment (ROI), service providers require tools that allow them to provision new services and then bill their customers for those services. In the past, service providers have accepted the notion that they would forever be limited to simple billing models, such flat-rate billing. We at Juniper Networks, Inc. believe that our latest hardware-based statistics collection tools provide the foundation that allows service providers to move beyond the restrictions of a flat-rate billing approach to deploy a multi-tiered billing model.

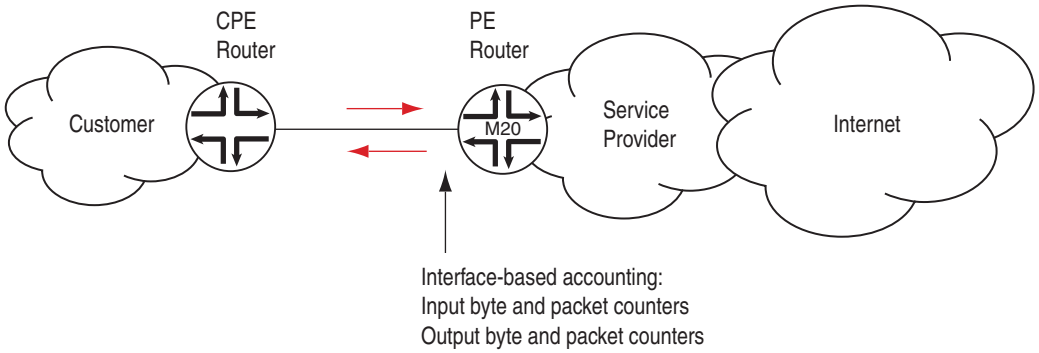
The first part of this paper takes a historical approach and describes the conventional interface-based accounting tools that service providers have traditionally used to collect network statistics. It also discusses mechanisms that allow these statistics to be moved from the router where they are gathered to the mediation systems that process the raw data to generate the information used for billing. The focus of this section is on how these traditional tools are supported by Juniper Networks® Internet backbone routers and JUNOS™ Internet software.

The second part of the paper discusses a number of new billing and reporting features that have been purpose-built by Juniper Networks to simplify and enhance the collection of accounting information. Filter-based accounting allows you, as a service provider, to configure customized packet filters to identify and then count user-defined traffic flows. Destination class usage (DCU) allows you to align your accounting policy with your routing policy and to have it dynamically adapt to rapidly changing routing environments. Because both filter-based accounting and DCU are implemented in hardware, they can be deployed at the edge *and* in the core of your network without significantly impacting packet-forwarding performance. Finally, we discuss accounting profiles that are designed to pre-process accounting information and to enhance the efficiency of transporting the statistics from the router to the mediation systems that generate the information used for billing.

Interface-based Accounting: The Historical Network Accounting Solution

Historically, service providers have collected accounting data from their routers by using an interface-based accounting approach. Interface-based accounting provides simply the volume of data that the customer transmits and the volume of data that the customer receives. (See Figure 1).

Figure 1: Interface-based Accounting Solution



In the Figure 1 illustration, the customer is connected to a local service provider that supports Internet access. Typically, interface-based accounting is performed at the inbound interface of the provider edge (PE) router or, if the provider delivers a managed routing service, at the

outbound interface of the customer premises equipment (CPE) router. The type of accounting information collected by interface-based accounting consists of input-packet and byte counters, as well as output-packet and byte counters.

There are a number of ways that JUNOS software allows service providers to access interface-based accounting information:

- Command-line interface (CLI), either through
  - Manual interaction or
  - Send and expect (send/expect) scripts,
- JUNOScript™ API, and
- Simple Network Management Protocol (SNMP).

Command-line Interface

The command-line interface (CLI), which automatically starts after the router finishes booting, is a straight-forward command interface that allows network administrators to perform various tasks, including configuring the JUNOS software, monitoring and troubleshooting the software, establishing network connectivity, and monitoring router hardware. The CLI may be accessed manually or through the use of send/expect scripts.

Manual Interaction

The CLI is accessed either manually, from the console, or through a remote network connection. Interface-based traffic statistics may be accessed by using the `show interfaces <interface-name> detail` command.

Send and Expect Scripts

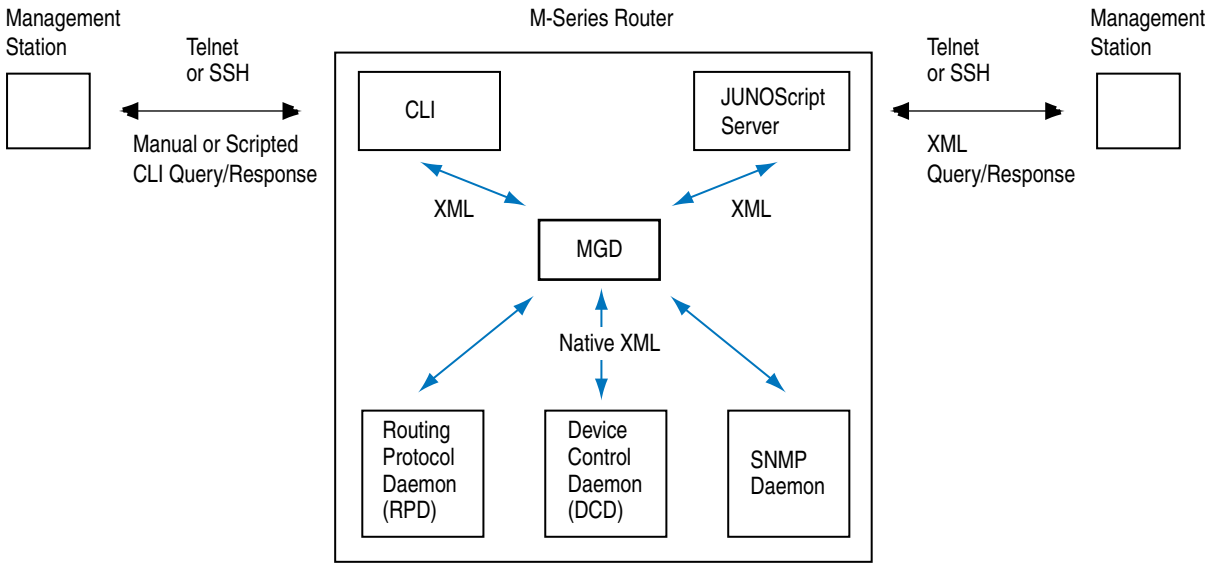
Interface-based accounting information can also be accessed by using scripting languages that network administrators use to automate router management. Although scripting languages are generally simple and comprehensive, they rely on CLI output from the user interface (UI) to support advanced configuration options and statistics collection. Unfortunately, managing the CLI programmatically can present problems for script developers, because the ASCII-based output has to be parsed for specific strings that are extracted and returned. If the ASCII output from a managed system changes (for example, from one software release to another), the logic that is used to parse that output must also be changed. Thus, maintaining scripts so that they continue to work and support new software releases removes the simplicity from the task and makes scripting a very complex, tedious, and error-prone task.

JUNOScript API

Extensible markup language (XML) allows users to write scripts that send and receive data in a structured format similar to the hypertext markup language (HTML). XML is text-based, easily generated, efficiently parsed by programs, unambiguous, extensible, self-describing, and

platform-independent. Scripts simply search the XML output for tags to locate specific attributes within a set of objects, rather than having to parse the output stream for specific ASCII strings that can potentially change from one software release to another. The JUNOScript interface provides an XML-based API for configuring and monitoring Juniper Networks routers programmatically.

Figure 2: The JUNOScript API



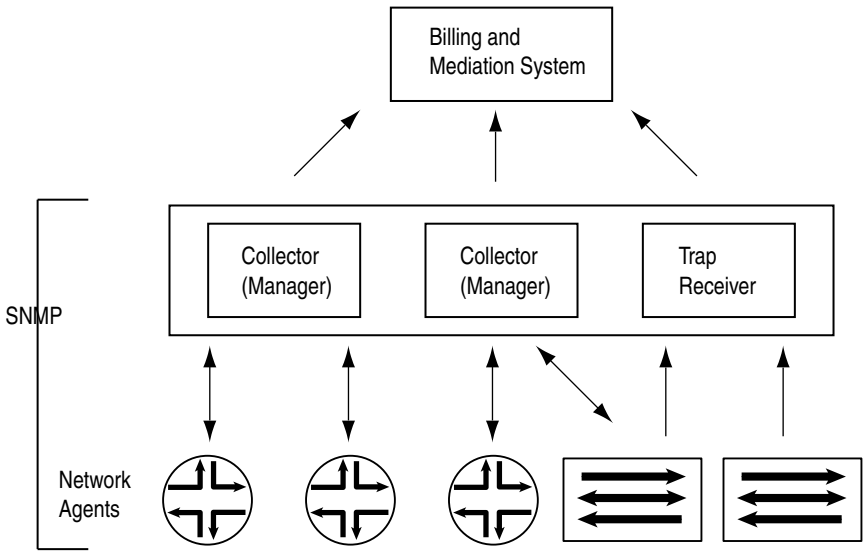
The JUNOS Internet software consists of a number of daemons, each running in its own protected memory space as an independent process over a common kernel. We have fully integrated XML into the management daemon (MGD), so that the MGD uses XML to communicate with other system daemons, as shown in Figure 2. This means that as new features are released they are immediately available through the JUNOScript API; therefore, you do not have to wait another six or nine months for the XML management interface to be developed. In fact, the functionality supported by the XML-based JUNOScript API delivers a superset of the functionality supported by the CLI. Juniper Networks' complete integration of XML into MGD is unequaled in the service provider marketplace.

The JUNOScript API makes the development and maintenance of scripts faster, easier, and more reliable, because the tags do not change from one software release to another, even if the CLI changes. XML gives us the flexibility to modify our CLI, yet to assure users that the scripts deployed in production networks will be simpler to manage and maintain.

Simple Network Management Protocol

The Simple Network Management Protocol (SNMP) allows a management system to collect accounting information from network elements by accessing the device's Management Information Base (MIB). The SNMP model consists of network information collectors (managers) that periodically poll network elements (agents) to gather accounting information. The raw information gathered by the collectors is correlated by the billing and mediation system, as Figure 3 shows.

Figure 3: SNMP Interaction with the Billing and Mediation System



The SNMP object that is most relevant to billing and accounting applications is the interface table (IfTable). Each line in the IfTable represents a router interface and contains detailed information about that interface:

- Interface type,
- Up or down status of the interface,
- Maximum transmission unit (MTU) supported over the interface,
- Counters containing the number of input bytes, input packets, output bytes, output packets, and errors.

SNMP was originally developed and deployed in the late 1980s. At that time, networks were commonly built using 10 Mbps Ethernet, Token Ring, DS-0, and T-1 interfaces, so 32-bit counters were more than adequate to track the amount of data being transmitted and received. Today, when 32-bit counters can be forced to wrap around in a relatively short period of time, one solution is to define shorter polling intervals, thereby accessing counters more frequently to ensure that the counters are not allowed to wrap around. Unfortunately, this approach is inadequate for high-performance routers that will be required to support OC-192c/STM-64 (SONET/SDH) and OC-768c/STM-256 interfaces or perhaps thousands of channelized interfaces. RFC 2233 (which JUNOS software currently supports) provides a solution to this problem by supporting a superset of the existing IfTable (the IFXTable) that uses 64-bit counters. Those 64-bit counters should be more than adequate to eliminate counter wrap-around issues for many years to come.

However, despite support for 64-bit counters, SNMP still has a number of limitations when it comes to supporting the bulk transfer of accounting information for high-capacity routing platforms:

- SNMP protocol data units (PDUs) make very inefficient use of bandwidth because they have a large overhead. To retrieve a single 32- or 64-bit counter, an SNMP management station needs to transmit an approximately 80-byte GetRequest PDU, and the managed agent needs to respond with an approximately 80-byte GetResponse PDU. This means that 160 bytes must be transmitted to transfer a single counter consisting of only 4 or 8 bytes of accounting information.

- SNMP is extremely slow, because it uses a zero-window transport protocol. This means that each request for information must receive a response before the next request for information can be made. Because SNMP uses User Datagram Protocol (UDP), not a variable-window transport protocol such as TCP, SNMP does not support multiple outstanding PDUs and is, therefore, challenged when asked to perform parallel tasks.
- SNMP consumes a significant amount of CPU resources. This is true for both the collectors that generate queries and for the managed elements that decode each query and then determine the specific information that the collector needs to retrieve.

An Ideal Accounting Solution

There are numerous ways that JUNOS Internet software allows service providers to access interface-based accounting information—through the CLI, send/expect scripts, JUNOScript API, and SNMP. Interface-based accounting simply provides raw data about the volume of traffic that each customer transmits and receives. Traditional voice carriers collect large amounts of billing information from each of their customers, allowing them to bill for differentiated services, such as the number of calls made, the destination number for each call, the time of day that each call was made, the long-distance carrier, and the number of minutes for each call. Compared with the amount of billing information that voice carriers can collect, the interface-based approach of billing customers solely on the volume of traffic, independent of the traffic type and destination, is an unsatisfactory solution for high-performance networks.

Let's begin by describing an ideal accounting solution from the perspective of Internet routing. The ideal accounting solution should provide the types of accounting information that allow you to move beyond the confines of a flat-rate billing system to the benefits of a multi-tiered billing approach. At the end of this paper, we will compare this ideal accounting solution with the various accounting tools that are supported by Juniper Networks routers and JUNOS software to see how close the Juniper Networks solution comes to the ideal model.

An ideal accounting solution possesses a number of distinct attributes:

- The accounting system should provide information about the total volume of data that each customer site transmits and receives.
- The model should provide information about the specific type of data transmitted (application, CoS, unicast, and multicast) to support billing on a multi-service network.
- The scheme should provide information about the source and destination of traffic flows to support source-sensitive and destination-sensitive class treatment.
- The accounting framework should allow the router that collects the data to assume some of the billing and mediation system's burden by allowing it to pre-process or aggregate the statistical data.
- The accounting strategy should provide a variety of different mechanisms to expose accounting data, such as SNMP, XML scripts, and FTP-retrievable flat files.
- The accounting system should support the transfer of statistics without placing an unreasonable load on the network or overwhelming the collection and mediation system.
- The approach should support the reliable transfer of accounting information from the router that collects the data to the management system that generate the information used for billing.

- The accounting strategy should adapt to a rapidly changing routing environment. This requirement obviates the use of static accounting templates that are unable to adapt to varying network conditions.
- The collection of accounting information must not impact the forwarding performance of the network so that the accounting solution can be deployed not only at the edges of the network, where services are created, but also in the core of the network on high-speed links.
- Finally, the ideal network accounting solution should offer you a variety of tools to give you the flexibility you need to design and deploy an accounting strategy that can be tailored to meet your specific requirements.

NetFlow: The Nascent Flow-Based Accounting Solution

Flow-based accounting was originally developed to enable service providers to collect accounting information for individual network services and to present customers with a different tariff for each service. An individual network flow can be broadly defined as a unidirectional sequence of packets between a given source and destination IP address. However, a finer granularity of flows can be identified by considering the IP protocol, the source and destination TCP/UDP ports, and the IP precedence bits.

The initial flow-based accounting solution, known as NetFlow, was an extension to the conventional route-lookup process. When the first packet of a traffic flow arrives at a router, the router's *flow cache* is searched using a hash key that is calculated across the fields of the packet header. If the system is unable to locate a matching cache entry, then a traditional longest-match IP routing table lookup is performed and the packet is forwarded. After the initial routing table lookup is performed, a new flow cache entry is created and the associated accounting data structures are defined. When a subsequent packet belonging to the same flow arrives at the router, the hash key is calculated, the flow cache entry is accessed, accounting is performed, the packet is forwarded using cached information, and the cache aging timer is reset. The flow-based accounting process sends an accounting record to the collector when each flow expires. The accounting record contains the statistics (counters and time stamps) that were collected while the flow was active. The flow cache entry has to expire or be terminated before the accounting record is made available to the billing and mediation system.

NetFlow supports two different types of accounting records. The first type of accounting record (V5) provides accounting information based on the source address, the destination address, the TCP or UDP classification, the Autonomous System (AS) fields, the starting time, and the ending time for each packet flow. The second type of accounting record (V8) provides the same accounting information as V5 records but introduces the ability to aggregate flow records to reduce the load on `cflowd` collectors.

While NetFlow was initially a very successful tool, it has a number of limitations when used for Internet backbone router applications or when flow-based accounting is performed on high-speed access links:

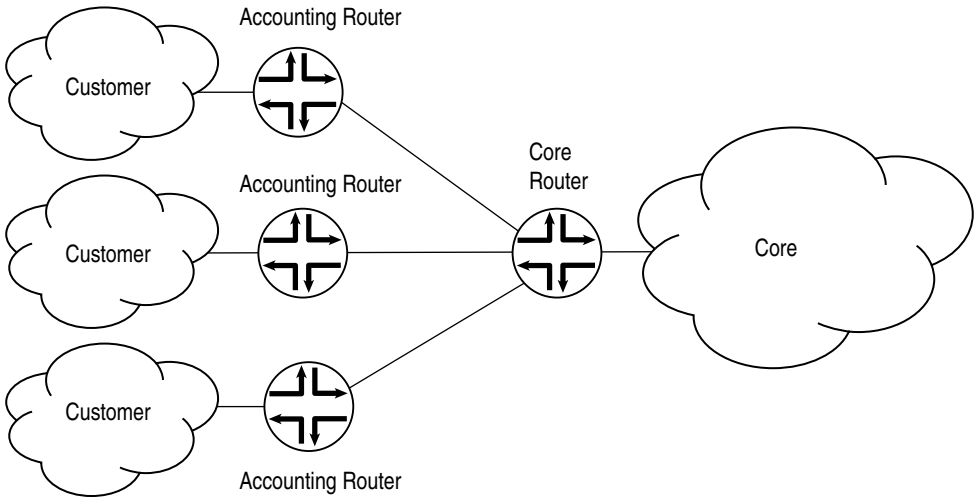
- Each time that network instability causes the routing table of a NetFlow router to change, it is necessary to flush the entire local routing cache and then rebuild it. This can have a severe impact on the CPU use of legacy routers. It is worth noting that Juniper Networks does not use route caching in our forwarding process. Instead, our Packet Forwarding Engine performs a traditional longest-match route lookup for each packet by using the high-performance capabilities of the Internet Processor II™ ASIC.

- NetFlow routers are required to maintain state for each packet flow. Maintaining large amounts of state can present problems, both for busy access routers at the edges of the network and for core router applications where each core router may be required to maintain state for potentially millions of flows.

Another challenge when performing NetFlow accounting on interfaces with speeds greater than OC-3/STM-1 is that *statistical sampling* is required to handle the increased traffic flow and to reduce CPU stress on both routers and collectors. However, the statistical sampling solution is still restricted to the edge of the network, because it has traditionally been performed in software. The result of performing statistical sampling in software is that it can severely impact the forwarding performance of legacy routers.

For these reasons, the NetFlow accounting mechanism is limited to lightly loaded interfaces at the edges of the network, and NetFlow cannot be successfully deployed in the core of the network. Network designers are often required to work around these limitations by deploying a large number of two-port accounting routers at the edges of their networks. (See Figure 4.)

Figure 4: Edge Accounting Routers



As Figure 4 illustrates, one port of each accounting router provides customer access, while the second port supports connectivity to the core of the network. Accounting routers perform only basic routing functions and are used primarily as accounting platforms. The need to deploy accounting routers to support flow-based billing functions adds to the overall capital cost of the network, as well as to the operational expense of managing additional routing systems.

When packets arrive at a core router, cacheless forwarding mechanisms can be used, because accounting information has already been collected at the edges of the network. Cacheless forwarding mechanisms can be viewed as an elementary version of the Juniper Networks Internet Processor II ASIC, because they perform route lookups in hardware but do not have the intelligence to support advanced packet processing features, such as filtering, rate limiting, statistics collection, and accounting.

Juniper Networks Response to Flow-based Accounting Solutions

While flow-based accounting is similar to traditional PBX-style accounting, it raises the question, "Should a service provider bill customers by tracking each traffic flow?" As a response to the scalability limitations of flow-based accounting solutions, we have developed a number of unique and powerful class-based accounting tools:

- Filter-based accounting,
- MPLS-based accounting, and
- Destination class usage (DCU) accounting.

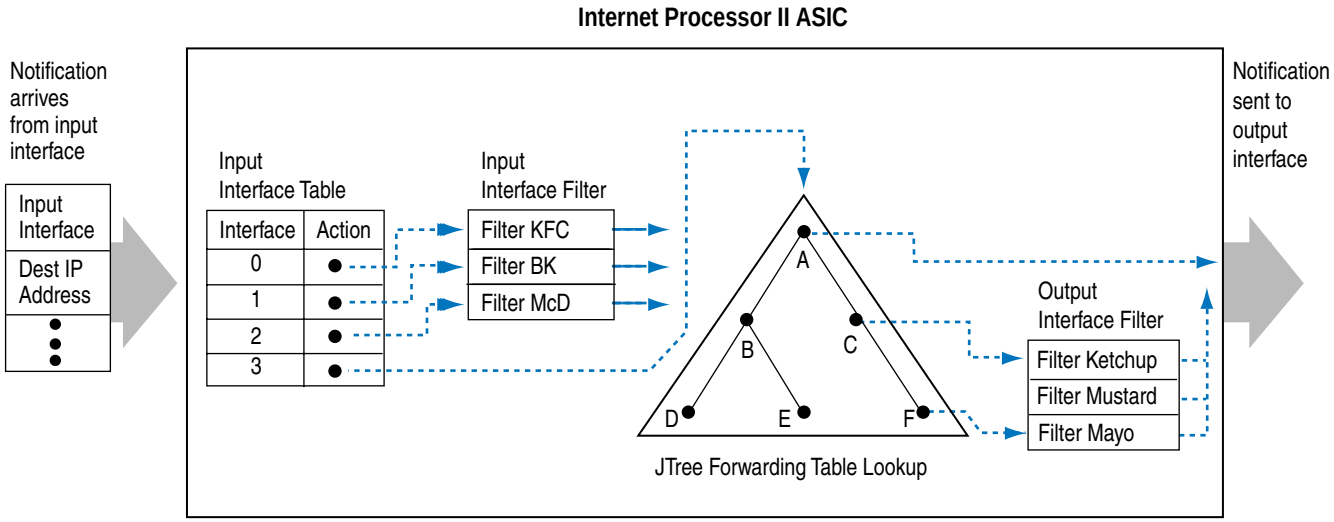
Each of these solutions is hardware-based and relies on the performance and flexibility of the Internet Processor II ASIC and its specialized interaction with JUNOS Internet software.

Filter-based Accounting

At the heart of the Packet Forwarding Engine in each Juniper Networks router is the Internet Processor II ASIC. The Internet Processor II ASIC supports not only industry-leading route-lookup performance, but also a number of advanced packet-processing features that can be enabled without significantly impacting the forwarding performance of our systems. These advanced packet-processing features include packet filtering, sampling, counting, and traffic policing, among others.

The performance of the Internet Processor II ASIC is achieved by using an extremely flexible method of programming the ASIC. (See Figure 5.) Network administrators write input or output packet filters to classify traffic in the manner that they are already accustomed to writing them. A compiler developed by Juniper Networks then automatically optimizes and compiles the filters for the Internet Processor II ASIC as part of the configuration process. Finally, the compiled filters are installed into the ASIC, where they run quickly and efficiently in hardware. The ability of the Internet Processor II ASIC to perform these functions while still maintaining stable forwarding performance allows you to deploy these tools both at the edges of the network, on high-speed access links, *and* in the core of your network.

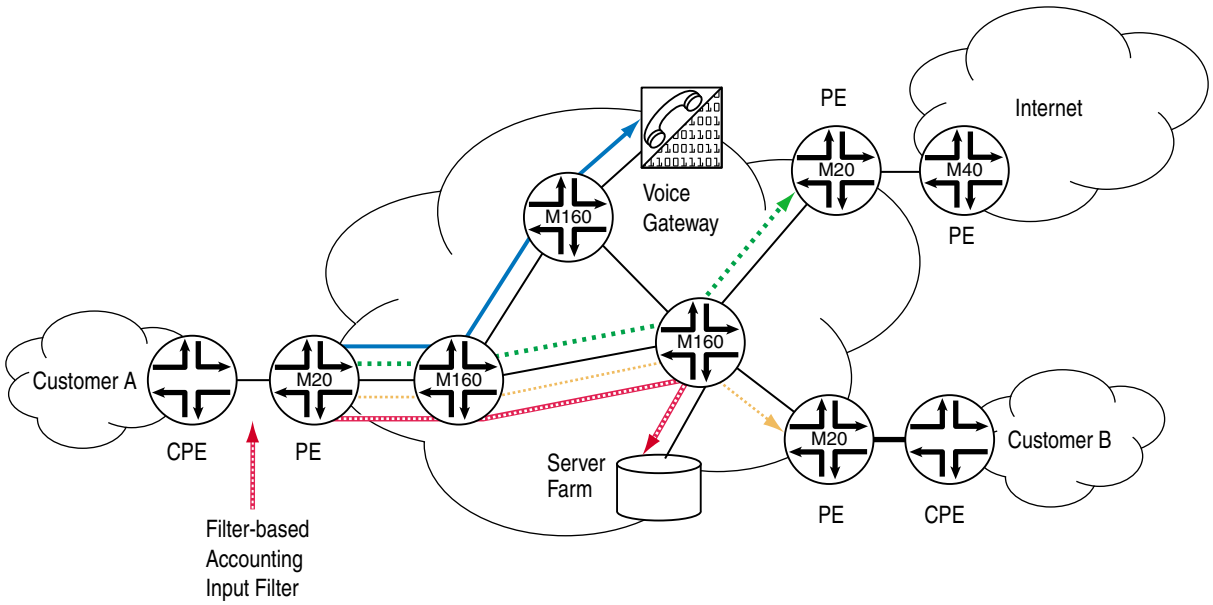
Figure 5: Internet Processor II ASIC Packet Processing Architecture



With respect to billing and accounting applications, JUNOS software and the Internet Processor II ASIC allow network administrators to configure filters that identify, accept, and count user-defined traffic classes. The criteria that can be used to specify a traffic class using packet filters is extremely broad and can include the source address, destination address, source and destination TCP, and UDP ports, for example. The system maintains 64-bit packet and byte counters that are available for subsequent retrieval through the CLI, send/expect scripts, the XML-based JUNOScript API, or SNMP.

Figure 6 illustrates how JUNOS software and the Internet Processor II ASIC can be used to support a typical filter-based accounting application.

Figure 6: Filter-based Accounting Example



In the example illustrated in Figure 6, the network administrator is interested in maintaining packet and byte counters for four specific traffic classes:

- The first class consists of Customer A's real-time traffic that is forwarded to the voice gateway. Because the provider offers expedited forwarding for real-time traffic, this traffic class can be identified by filtering on the DiffServ byte in the packet header.
- The second class contains Customer A's best-effort traffic that remains within the provider's AS and is destined for the server farm. This traffic class can be identified by filtering on the server farm's destination IP address.
- The third class is composed of Customer A's best-effort traffic that exits the AS and is destined for Customer B. This traffic class can be identified by filtering on Customer B's destination IP address.
- The fourth class carries Customer A's best-effort traffic that exits the AS and is destined for the public Internet. This traffic class consists of all traffic that does not match one of the three previous filter conditions.

As discussed earlier, packet and byte counters for these packet filters can be accessed using the CLI, send/expect scripts, JUNOScript API, or SNMP.

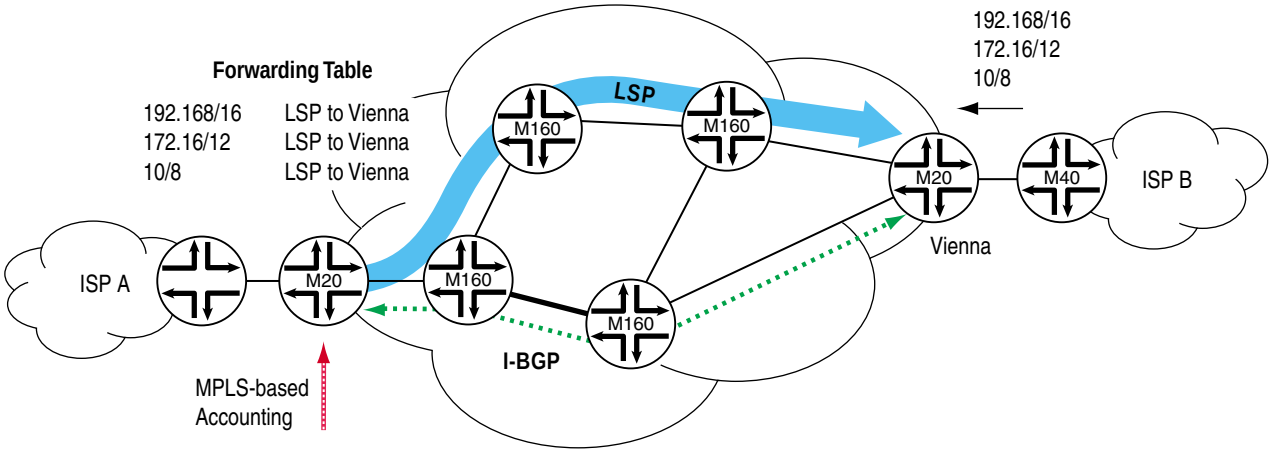
Although filter-based accounting provides an extremely useful tool, it has limitations because network administrators must know in advance what traffic they want to track and must have the skill to write a packet filter to identify and count that particular traffic. For example, if you want to count the volume of traffic flowing to each of your subscribers, every time that your network administrators add or remove a subscriber, they have to update the filters on all of the routers that perform accounting functions in your network. While packet filters can be maintained by using scripts, this approach does not really scale.

In production networks, filter-based accounting is useful for performing baseline classification of the *class* of traffic that is being transmitted by a customer. Filter-based accounting is an excellent tool for tracking implicit traffic classes (based on CoS or IP precedence), as well as for counting the volume of traffic that is sent to *golden* networks. A golden network is a network that never changes, such as a server farm.

MPLS-based Accounting

Multiprotocol Label Switching (MPLS) has several advantages, such as facilitating traffic engineering, supporting the delivery of Layer 2 and Layer 3 VPNs, and providing network resiliency. As Figure 7 shows, there is also an interesting benefit supported by MPLS with respect to billing and accounting.

Figure 7: MPLS-based Accounting



In this example, the network administrator collects statistics for the traffic that flows across the label-switched path (LSP) to the router identified as Vienna. This is an appealing solution, because ISP B's prefixes are dynamically mapped to the LSP so the accounting solution can easily scale from one prefix to more than 100,000 prefixes, either within or downstream from ISP B. This makes it very easy for network administrators to manage accounting, because the mapping of prefixes to an LSP is dynamic, not static. In addition to a wide range of LSP statistics, packet and byte counters for traffic traversing the LSP can be accessed using the CLI, send/expect scripts, JUNOScript API, or SNMP.

Destination Class Usage (DCU) Accounting

Despite the benefits of MPLS as a dynamic accounting tool, we understand that some service providers do not want to deploy MPLS in their networks. Nonetheless, they still need accounting tools that allow them to maintain packet and byte counters based on the ingress and egress points for traffic traversing their networks. The ingress point for a traffic class can be easily identified by the input interface. Egress points can be identified using the Juniper Networks destination class usage (DCU) tool that groups destination prefixes into one or more disjointed sets and then maintains per-interface packet and byte counters for each set.

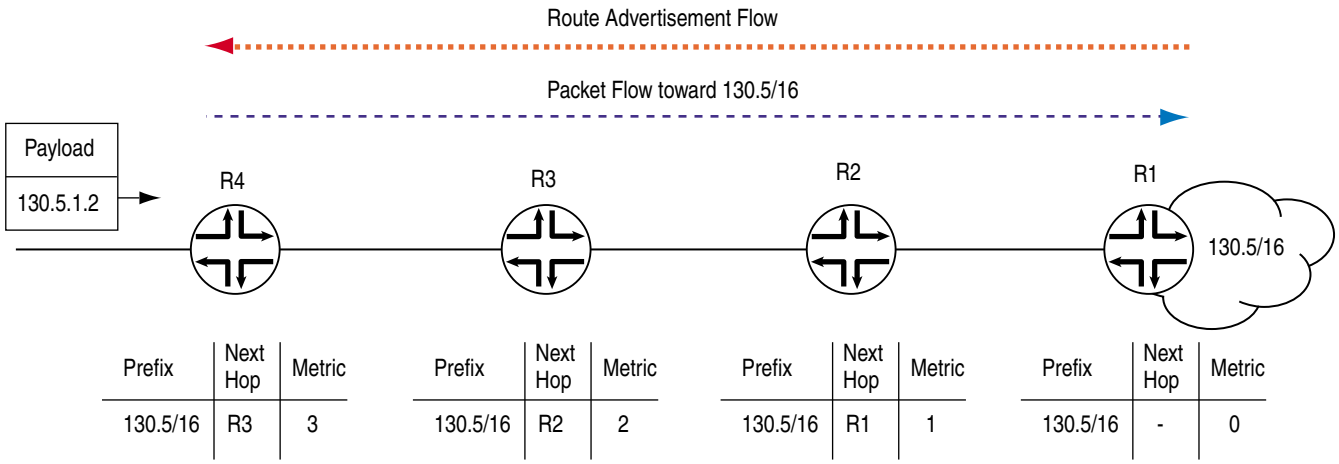
To help clarify the benefits of the Juniper Networks DCU feature, our discussion will focus on three important issues related to this feature:

- Routing policy,
- The DCU operational model, and
- The impact of DCU on packet-forwarding performance.

Routing Policy

A route is a pointer to a destination network. With the receipt of a routing advertisement, a router learns not only a route prefix but also a number of attributes associated with the route. Routing protocols such as Open Shortest Path First (OSPF) and Intermediate System to Intermediate System (IS-IS) typically carry a limited number of route attributes; whereas, the Border Gateway Protocol (BGP) supports the transmission of numerous route attributes. It is interesting to note that route advertisements flow in the direction opposite of user data traffic flowing to the advertised prefix, as Figure 8 shows.

Figure 8: Routing Advertisements and Data Traffic Flow in Opposite Directions



Routing policy allows network administrators to create a wide variety of policies and apply them as routes move between routing protocols and the routing table within a router. Typically, routing policy is used to control the import or export of routes to other routers (or ASes). Additionally, routing policy can be used to change the attributes associated with a route, such as the metric, the AS-Path, or the BGP community.

Implementing routing policy filters to group routes, so that a common action can be applied to all routes that match the filter condition, is fundamentally different from implementing firewall filters. In the case of firewall filters, a packet filter is used to classify user *data packets* based on an examination of packet header fields. If a packet matches the conditions of a firewall filter, the user packet can be accepted, discarded, counted, sampled, logged, policed, or rate-limited. In the case of routing policy, a routing policy filter is used to classify (or colorize) *routes* based on an examination of attributes that are advertised with the route. These attributes can include the OSPF area ID or tag field values, the IS-IS level number, or the BGP AS-Path, community, local preference, or origin attributes. All routes that are assigned the same *color*, as it is called, are treated the same with respect to the routing policy action. Routing policy can determine several actions.

- Accept and propagate all matching routes.
- Reject and do not propagate all matching routes.
- Modify specific route attributes for all matching routes before importing the route from or exporting the route to a routing peer.

By altering routing policy filters, you can change the flow of packets through your network, because modifying routing policy filters has a fundamental impact on the distribution of routing information.

BGP was originally designed to be a policy-driven routing protocol. Initially, service providers wrote routing policy filters to colorize routes based on route prefixes. However, as the Internet continued its rapid growth, the task of maintaining the filters, adding and removing prefixes as the Internet constantly changed, and redistributing the updated filter lists simply did not scale. This caused service providers to use the less granular, but more manageable, approach of using filters based on AS paths. Then, as the competitive landscape caused large ISPs to acquire smaller ISPs and it became common for ISPs to have more than one AS number, the task of

maintaining AS-based policy filters became more complicated. Today, many service providers are employing the even less granular approach of using BGP Communities and BGP Extended Communities to colorize their routes.

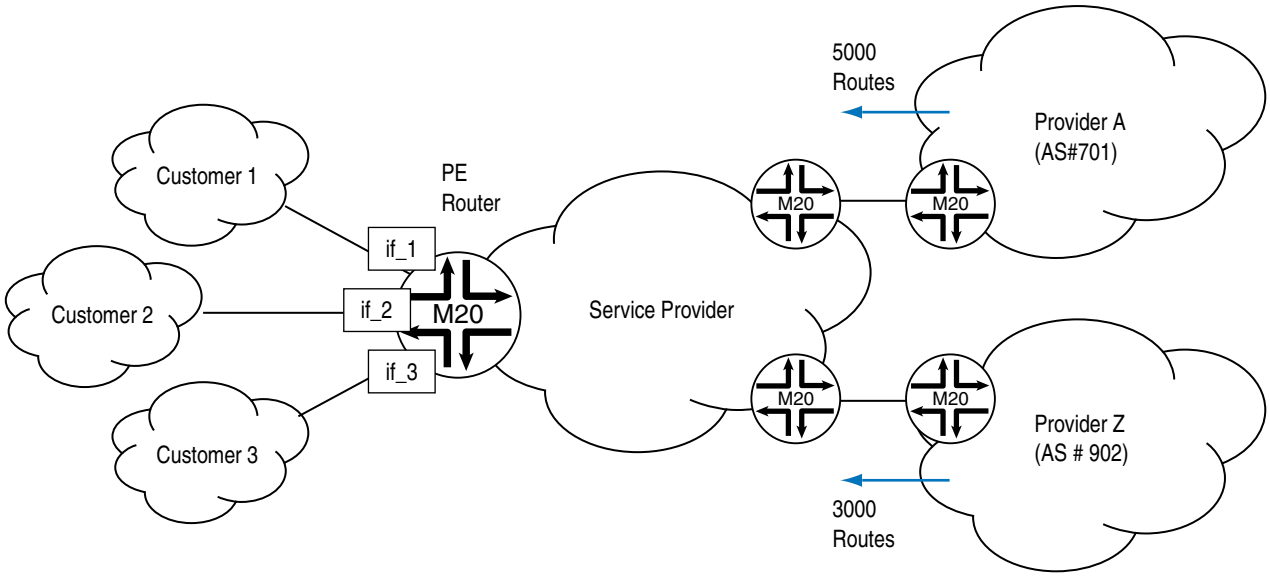
Destination Class Usage Operational Model

Destination class usage (DCU) is a policy-based accounting mechanism that is designed to support accounting for up to 16 different traffic classes. Cell phone providers generally have a maximum of 6 or 7 different tariffs—local, state, national, international, peak, non-peak, and weekends. DCU allows a service provider to

- Define up to a maximum of 16 unique service classes (or billing buckets) per router,
- Use local routing policy to associate routes in the forwarding table with one or more of these 16 buckets, and then
- Maintain per-interface packet and byte counters to tally the traffic that matches each of these buckets.

Figure 9 provides an example in which a service provider purchases Internet transit service from two different providers—Provider A (AS number 701) and Provider Z (AS number 902). The service provider wishes to bill its customers using one tariff for traffic using Provider A and a different tariff for traffic using Provider Z.

Figure 9: Destination Class Usage (DCU) Example



The service provider writes an AS-based or Community-based routing policy filter that states that all routes received from Provider A should be associated with DCU bucket number 1, and that all routes received from Provider Z should be associated with DCU bucket number 16. If Provider A advertises 5000 routes, the routing policy tells the Internet Processor II ASIC to set the first bit in the 16-bit DCU bit mask associated with each of these 5000 routes to 1 (thereby linking each of the 5000 routes received from Provider A with DCU bucket number 1). If Provider Z advertises 3000 routes, the routing policy tells the Internet Processor II ASIC to set the sixteenth bit in the 16-bit DCU bit mask associated with each of these 3000 routes to one

(thereby linking each of the 3000 routes received from Provider Z with DCU bucket number 16). Now, assume that DCU is enabled on If_1 and If_3 of the PE router, but not If_2 of the PE router.

We now consider examples that describe how DCU counts packets arriving at the PE router from Customer 1, Customer 2, and Customer 3.

- Assume that a packet arrives on If_1 that is destined for a subnet that the service provider has learned from Provider Z. The Internet Processor II ASIC performs a standard route lookup in its forwarding table. Because the bit mask associated with the route has bit 16 set to 1, the Internet Processor II increments the bucket number 16 counter associated with If_1. However, the bucket number 16 counters associated with other input interfaces are not incremented.
- Assume that a packet arrives on If_2 that is destined for a subnet that the service provider has learned from Provider A. The Internet Processor II ASIC performs a standard route lookup in its forwarding table. Because DCU is not configured for If_2, the Internet Processor II ASIC does not increment any counters.
- Assume that a packet arrives on If_3 that is destined for a subnet that the service provider has learned from Provider A. The Internet Processor II ASIC performs a standard route lookup in its forwarding table. Because the bit mask attached to the route has bit 1 set to 1, the Internet Processor II increments the bucket number 1 counter associated with If_3. However, the bucket number 1 counters associated with other input interfaces are not incremented.

Typically, each route is associated with a single bucket. However, depending on the specific routing policy filters that are configured for a system, a route may be linked with multiple buckets. Thus, the bit mask associated with any given route may have all 16 bits cleared to zero, one bit set to one, or multiple bits set to one.

Impact on Packet Forwarding Performance

DCU accounting has a minimal impact on the packet-forwarding performance of the Internet Processor II ASIC, because it does not significantly increase the amount of processing that must be performed on each packet. Inasmuch as the Internet Processor II ASIC does not have direct access to AS path or BGP Community attributes, the JUNOS software uses routing policy to create the DCU bit masks that are associated with the prefixes in the forwarding table. When a packet is processed by the Internet Processor II ASIC, it performs a standard route lookup in its forwarding table, identifies the traffic classes associated with the route by examining the DCU bit mask, and then uses the input interface number carried in the packet notification to increment the appropriate packet and byte interface counters.

Figure 10: Internet Processor II ASIC Support for DCU

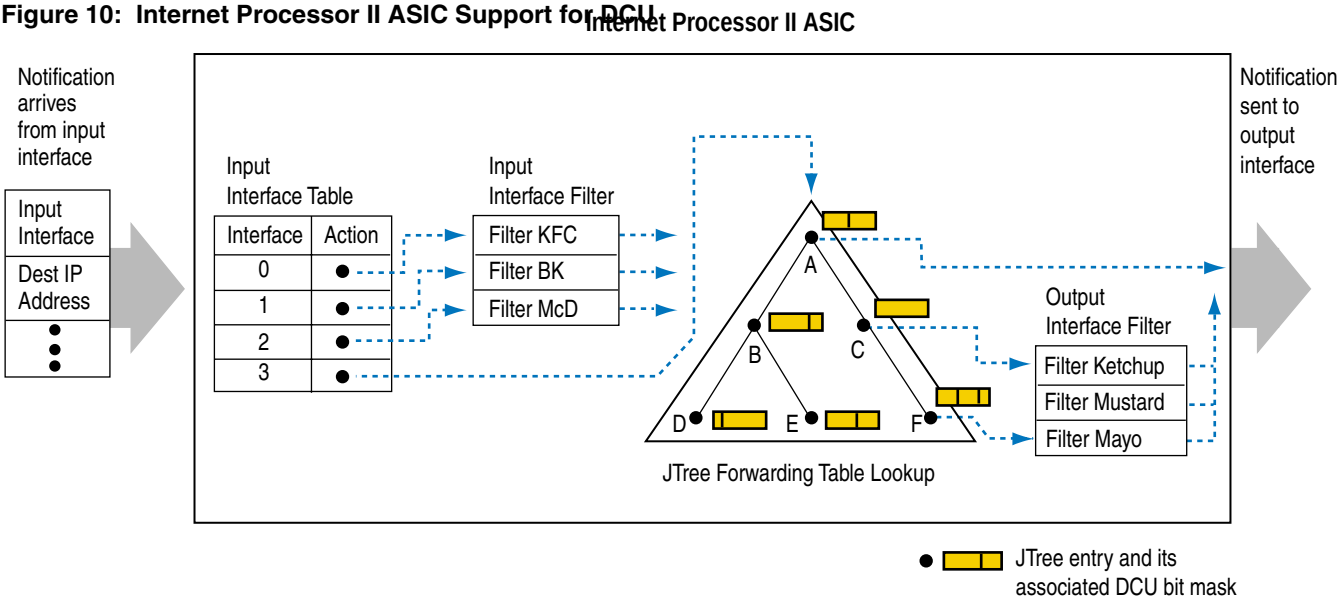


Figure 10 illustrates how the Internet Processor II ASIC supports DCU accounting. The only change to the forwarding path through the Internet Processor II ASIC is the addition of a bit mask that is associated with each route in its forwarding table. Enabling DCU does not significantly increase the complexity of the Internet Processor II ASIC's hardware-based processing path, other than requiring a simple bit mask examination and potential counter incrementation.

Let us now consider some examples that describe how the Internet Processor II ASIC processes packet notifications to support DCU:

- Assume that a packet arrives on input interface number 1 and that the destination IP address matches prefix A in its forwarding table. After the input filter program *Filter BK* accepts the packet and the route lookup is performed, the Internet Processor II ASIC examines prefix A's bit mask and discovers that the packet matches DCU bucket number 8. The Internet Processor II ASIC increments interface number 1's DCU bucket number 8 packet and byte counters. Finally the packet is forwarded to its output interface.
- Assume that a packet arrives on input interface number 2 and that the destination IP address matches prefix C in its forwarding table. After the input filter program *Filter McD* accepts the packet and the route lookup is performed, the Internet Processor II ASIC examines prefix C's bit mask and discovers that the packet does not match any of the DCU buckets. Because the Internet Processor II ASIC does not have to increment any counters for this packet, it continues to process the packet by executing the output filter program *Filter Ketchup.*
- Assume that a packet arrives on input interface number 3 and that the destination IP address matches Prefix F in its forwarding table. After the route lookup is performed, the Internet Processor II ASIC examines prefix F's bit mask and discovers that the packet matches DCU bucket number 6 and DCU bucket number 14. The Internet Processor II ASIC increments interface number 3's DCU bucket number 6 and DCU bucket number 14 packet and byte counters. After incrementing these counters, the Internet Processor II ASIC continues processing the packet by executing the output filter program *Filter Mayo.*

Administratively Scalable Billing and Accounting Solutions

Successful Internet services are required to scale along two dimensions. The first dimension is technical scaling, which is concerned with the challenge of creating systems that can perform increasingly complex packet-processing functions without significantly impacting the system's forwarding performance. The second dimension is administrative scalability, which is concerned with providing the tools that you need to provision new services and then bill your customers for those services. Any successful service offering requires scaling along both dimensions simply because you must have a mechanism that allows you to bill for a new service before you can sell it to your customers.

Next, we will examine the administrative scaling capabilities of Juniper Networks' billing and accounting solutions:

- Interface-based accounting
- Filter-based accounting
- MPLS-based accounting
- Destination class usage (DCU) accounting

Interface-based Accounting

From an administrative point of view, interface-based accounting is very adaptive to dynamically changing traffic patterns. Once the customer interface is provisioned, the router automatically maintains statistics that describe the volume of data that the customer transmits and receives over a given interface. While interface-based accounting is certainly the easiest solution to deploy, the type of accounting information provided is inadequate to support the requirements of a multi-tiered billing system.

Filter-based Accounting

Filter-based accounting scales nicely with respect to administration as long as the prefix information does not change. For example, filter-based accounting works extremely well when a service provider wants to count traffic to any destination based on the DiffServ byte (for example, to distinguish between real-time traffic and best-effort data traffic) or when tracking traffic addressed to golden networks, where destination prefixes do not change. Juniper Networks' filter-based accounting is a very attractive solution for these applications, because counting filters can be enabled on our routers without significantly impacting packet-forwarding performance.

However, filter-based accounting does not scale with respect to administration when network prefixes are constantly changing. The scalability issues are related to the requirement that service providers know the specific prefixes that they need to examine if they want to write the required counting filters, the challenges of monitoring route additions and withdrawals as they occur in other service provider networks, and the need to constantly update and deploy revised packet counting filters to reflect these changes. Filter-based accounting requires a significant amount of human intervention to align accounting policy with a constantly evolving routing environment.

MPLS-based Accounting

The dynamic nature of prefix-to-LSP mapping allows MPLS-based accounting to exhibit some of the administrative scalability provided by DCU. If a BGP speaker announces a certain set of routes, then the LSP to that BGP speaker will be used to forward traffic to those prefixes. Routing policy gives service providers a tremendous amount of control in selecting the

prefixes that are mapped to each LSP by controlling which routes are advertised by each BGP speaker. Consequently, MPLS-based accounting tools provide administrative scalability in both single label-stack environments (for traffic engineering) and in multiple label-stack environments (for MPLS-based VPNs).

DCU Accounting

DCU accounting has been specifically designed to scale administratively when prefix information is in a state of flux. DCU supports administrative scalability because accounting is directly aligned with the configuration of routing policy. This is an important advantage, because DCU does not require you to spend significant amounts of time synchronizing routing policy with filtering or accounting policies. Thus, you do not have to constantly update your accounting policies as new routes are added, old routes are withdrawn, or routes are learned over new interfaces when network topology changes. Whenever a route advertisement changes, the accounting policy changes dynamically, because DCU accounting is linked to the route-lookup process, not a specific router interface. Juniper Networks stands alone in its ability to deliver this enhanced accounting capability due to the processing flexibility and forwarding performance of the Internet Processor II ASIC.

The existing limitations of DCU accounting are that traffic can be assigned to an accounting bucket based only on the destination address carried in the packet header, and that JUNOS software does not currently support the ability to keep track of traffic flowing in the return direction. However, the programmability of the Internet Processor II ASIC, along with the flexibility of JUNOS software, allows us to continue to enhance the capabilities of DCU and allow it to function as a more generalized packet-class-treatment tool.

Juniper Networks Accounting Profiles

Earlier in this paper, we discussed some of the limitations of using SNMP to transfer large amounts of accounting information from the routers that collect statistics to the network management systems that process the statistics to generate information for billing. To reiterate, there are three primary limitations.

- The format of SNMP PDUs requires a significant amount of overhead to transmit a small 32- or 64-bit counter value,
- SNMP is a zero-window application-level transport protocol, which makes it very slow with respect to the transfer of data between agents and managers, and
- The generation of SNMP PDUs consumes a significant amount of CPU resources on both management platforms and managed elements.

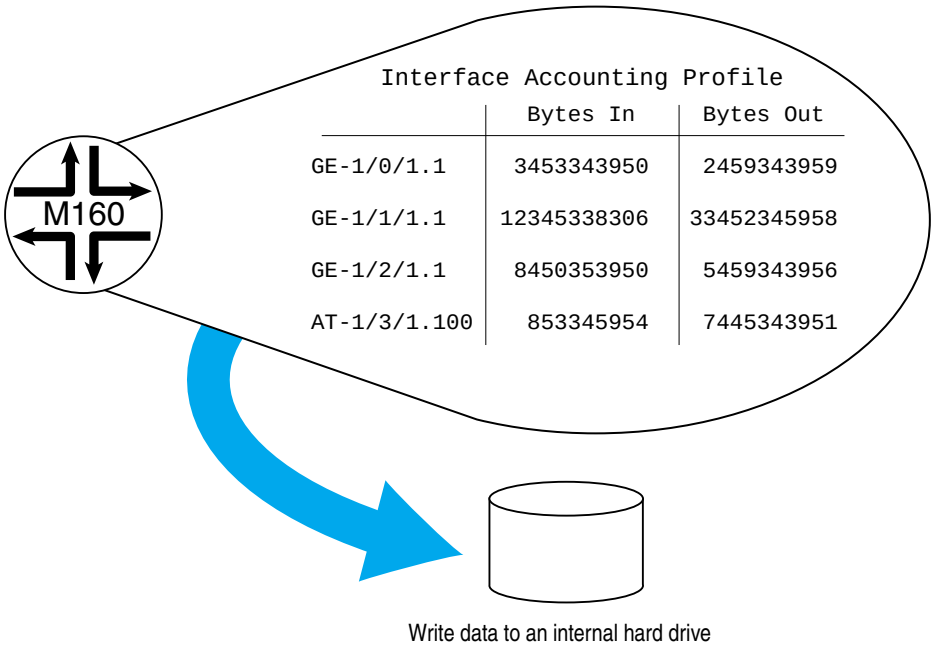
In the early 1990s, when service provider routers had only a few interfaces, SNMP was an acceptable tool to collect billing and accounting information. However, for high-capacity aggregation routers that can have potentially thousands of logical interfaces, SNMP is a very inefficient tool for the transfer of bulk data for several critical reasons:

- To support a single device with thousands of interfaces, the SNMP model requires a large number of SNMP GetRequest and GetResponse PDUs, a significant amount of CPU processing on both the management station and managed elements, large quantities of bandwidth to transmit these messages, and an alarming amount of correlation to make sense of all of the data.

- There are also the repercussions of lost accounting information when an SNMP collector cannot communicate with managed elements due to network connectivity problems. The inability to transmit and receive GetRequest and GetResponse PDUs means that accounting information cannot be collected, the counters on managed elements continue to increment and wrap around. Therefore, you cannot bill for services.

To address the limitations of the SNMP model when supporting high-performance routing platforms, Juniper Networks has developed what we call *accounting profiles*. Accounting profiles provide a mechanism that allows network administrators to configure a router to collect accounting information and write it to a flat file on a local disk with an appended time stamp. Then, when appropriate, network administrators can retrieve the accounting files and import the pre-processed data into their billing and mediation system. For customers who prefer the push approach to the poll-and-pull method, accounting profiles can also be configured to push the flat file to a specific billing or collection station. (See Figure 11.)

Figure 11: Interface Accounting Profile



Accounting profiles define the characteristics of the accounting data that will be collected, including the collection interval, the name of the file that contains the statistics, the maximum number of bytes per file, the number of rolling files that should be maintained, and the specific fields and counter names that should be collected. JUNOS software supports three types of accounting profiles:

- An interface profile collects specific error and statistical information for router interfaces or sub-interfaces.
- A filter profile supports the collection of packet and byte counters for the specific firewall filter counter names specified.
- A DCU profile collects packet and byte counters for specific DCU interface counters.

Configuring accounting profiles overcomes many of the limitations of the SNMP model when attempting to manage high-capacity access routers.

- The elimination of an extremely large number of SNMP PDUs and their associated overhead allows bandwidth to be used more efficiently for revenue-generating services.
- The CPU resources that are required to generate and respond to SNMP messages on management platforms and managed systems can be used for other tasks.
- The sluggish performance of SNMP's zero-window protocol can be overcome by the use of variable-window protocols that run on top of TCP, such as FTP or secure shell copy (SCP), which are specifically designed for the reliable transport of bulk data.
- Finally, if the management system loses communication with a router that is configured to maintain an accounting profile, the collector can fetch the accounting information from the network element when connectivity is re-established. This minimizes the chance of losing billing and accounting information.

Conclusion

In this paper, we described an ideal accounting solution from the perspective of Internet routing. It is clear that a single tool cannot satisfy the requirements of every accounting application, considering the various trade-offs that need to be made when collecting statistical information for diverse accounting applications deployed in different parts of the network (such as the volume of information to be transmitted, the need for reliability, the need to summarize or pre-process data vs. the immediate transmission of raw data, interface statistics vs. aggregated statistics, counting each packet vs. statistical sampling, temporary storage on the router and then bulk transfer of data vs. immediate data transfer).

Perhaps the only way to achieve the goal of delivering the ideal accounting solution is to provide a portfolio of tools, each of which is specifically designed to support some of the attributes of the ideal accounting solution. The more flexibility that a vendor can provide, the greater the number of tools that you can use to gather network statistics and move those statistics to management systems. A truly flexible approach enables you to consider the specific requirements for each accounting application, and then to select the best approach or combination of approaches from a portfolio of accounting tools.

Table 1 compares the attributes of the ideal accounting solution against each of the statistics-collection tools supported by Juniper Networks routers and JUNOS Internet software.

Table 1: Accounting Information Collection

The Ideal Accounting Solution	Juniper Networks Accounting Tools			
	Interface Accounting	Filter-based Accounting	MPLS-based Accounting	DCU Accounting
Traffic volume statistics	Yes	Yes	Yes	Yes
Traffic type statistics		Yes		Yes
Traffic source statistics		Yes		
Traffic destination statistics		Yes	Yes	Yes
Adaptability to dynamic routing environment			Good	Excellent
Impact on packet-forwarding performance	Minimal	Minimal	Minimal	Minimal

Table 2 compares the attributes of the ideal accounting solution against each of the mechanisms supported by Juniper Networks routers and JUNOS software for the transport of accounting information to management systems. Despite the limitations of SNMP, we continue to make accounting data available using the SNMP interface so service providers have the freedom to make their own choices, and so we can continue to support the majority of existing network management systems.

Table 2: Transfer of Accounting Information

The Ideal Accounting Solution	Send / Expect Scripts, CLI	JUNOScript API	SNMP	Accounting Profiles
Pre-processing or aggregation of statistics	No	No	No	Yes
Transfer of statistics without placing a significant load on the network	Yes	Yes	No	Yes
Reliable transfer of statistics to management stations	Yes	Yes	No	Yes

We believe that our portfolio of hardware-based accounting tools comes very close to delivering the features of the ideal accounting solution. You no longer need to think about solving your accounting problems in a constrained manner, because viable alternatives to traditional software-based tools are available—today. The application of highly integrated silicon to the accounting problem allows Juniper Networks to change the fundamental nature of what is technically feasible from accounting tools. It is the power and flexibility of JUNOS software and the Internet Processor II ASIC that permit us to continue evolving these tools without forcing our customers to constantly upgrade their hardware to take advantage of our enhanced solutions.

Part Number: 200010-001 07/01