MANAGING PEER-TO-PEER TRAFFIC – BEYOND DOCSIS 1.1

John R. Pickens, PhD
Greg Hutterer
Allot Communications, Inc.

Abstract

Managing traffic rates and assuring that networking resources are fairly offered and consumed is perhaps the penultimate requirement and opportunity for HFC cable networks. Failures translate rapidly to customer dissatisfaction and lost revenue. Current mechanisms such as DOCSIS™ 1.1 and PacketCable™ DQOS provide strong traffic management functionality for select specified applications (e.g. VOIP). However the rapid demand for new applications and services (e.g. peer-to-peer), coupled with the long cycle-time of specification, implementation, testing, and deployment is rapidly bringing networks to their knees. New mechanisms such as in-line flow classification and application signature detection enable operators to quickly understand and adapt to new application paradigms (e.g. peer-to-peer), and fulfill rapidly changing subscriber demand. New tools and interfaces are needed to accelerate service revenue and enhance customer satisfaction.

BACKGROUND

Few network designers anticipated the exponentially growing traffic levels and ever-increasing, almost viral, portfolios of applications, architectures and protocols seen in broadband networks today. In particular, the original application traffic assumptions driving design of HFC cable networks have been vastly exceeded. The end-user cable environment is rife with subscriber-popular, bandwidth-hungry applications, each one vying for its share of the available HFC first-mile bandwidth.

Peer-to-Peer (P2P) file sharing in particular has emerged as a highly popular IP technology, primarily among home users. The rapid rise of protocols such as KaZaA, Morpheus, and Gnutella allow virtually every computer to become a server, freely sharing enormous audio and video files at will among users of an uncontrolled global community.

Community	Users
FastTrack	4,464,221
iMesh	1,421,256
eDonkey	582,030
Overnet	315,592
DirectConnect	151,898
Blubster	93,883
Gnutella	83,439

Figure 1 - P2P Communities
(**www.slyck.com**)

The largest P2P file sharing communities have millions of users – The FastTrack community has nearly 5 million users (FastTrack uses KaZaA Lite, Grokster, KaZaA, and iMesh clients). No ISP or MSO can fail to notice the impact of Peer-to-Peer file sharing.

P2P file sharing applications can consume the majority of the total bandwidth, even with only a few subscribers active, making this issue a major concern for cable providers.

The following issues are increasingly being noted by operators:

- P2P generates high traffic loads - delaying more urgent traffic of other subscribers and negatively affecting customer satisfaction.

- P2P traffic consumes even higher upstream bandwidth resources. As knowledge of each new subscriber node is propogated through the network, more and more peer nodes request uploads, thus exponentially increasing upstream traffic.

- P2P increases operator costs by forcing WAN capacity upgrades and HFC capacity upgrades through reducing CMTS subscriber density and splitting fiber nodes.

An opportunity exists for MSOs to manage the bandwidth crisis imposed by new application paradigms like P2P file sharing and also reap new service revenue by offering strong and flexible Quality of Service functions to subscribers. Not only the traditional applications (web, email) and the operator applications (VOIP) can be serviced, but new application paradigms such as peer to peer (from KaZaA to Grid computing), broadcast streaming (from MPEG video to internet radio), and two-way voice and video conferencing (from PacketCable voice to Voice/Video Instant Messaging) can be serviced as well. Also the operator can detect new applications as they are activated by subscribers, and can implement policies with respect to traffic prioritization or even traffic-blocking.

APPLICATION PARADIGMS

Original Broadband Application Paradigm

When broadband networks were first conceived, designed, and deployed, the assumed applications model was occasional web browsing and electronic mail, with infrequent activation of bandwidth hungry applications like file transfer. This model was primarily downstream assymetrical with intermittently active users.

This assumed applications model has evolved to include new planned subscription services such as symmetrical-bandwidth cable telephony. Both deployed networks and DOCSIS [1,2] and PacketCable [3,4] protocol standards have been carefully architected to support these sorts of carefully designed applications.

This initial set of applications and services paradigms is server centric – web servers, email servers, and VOIP soft-switch servers. The server model is strong in its ability to manage the scaling of services, centralize provisioning, and support trusted authentication and authorization schemes.

But, as users, and especially developers, become aware of the high-bandwidth, low-latency, and always-on attributes of broadband, new application paradigms rapidly emerge. Subscribers begin adopting the applications. Because of the bandwidth-intensive traffic profiles of these new applications, the existing best-effort QoS mechanisms fall apart.

New Broadband Application Paradigms

"The media is the message" – Marshall McLuen, 1967 ***[5]****.*

Client-server is not the only (or even best) model for distributed processing. As the

speed of the communications link rises, latency drops, and peer-to-peer reachability becomes pervasive, other distributed processing models become possible.

Peer to peer (P2P) has achieved recent notoriety, especially for its seemingly unconstrained penchant for bandwidth consumption. P2P is notable in that there is often no central point of control (although there are some hybrid P2P with super server architectures for scalability). The list of P2P applications is long, and new applications are coming online every day.

Is the appearance of P2P a surprise? It should not be. P2P is a member of a class of new distributed applications model types, each of which are enabled in the new world of always-on, high-bandwidth networking.

There exist at least six distributed computing models, each of which is enabled by Broadband **[6]**:

- Ad hoc distributed computing model – no specific architecture constraints. The applications developer has a networking API at his disposal and is free to generate any partitioning of functions using any protocol.

- Remote Procedure Call model – the application uses a procedure call interface. The procedure function name and all calling parameters are shipped to the remote node, and the application waits until a procedure return is invoked with the return result.

- Remote Evaluation model – fragments of applications are moved to the remote system on which the data is contained. The application works on local data without incurring any cross-network latency. The remote environment may require call-outs or data-sharing facilities to access data from the invoking environment.

- Remote compute cluster model – all file, database, and computational resources are collocated on a remote high speed low latency network. What flows between the client and the compute cluster is keyboard/mouse input and screen output (bitmapped buffers or 2D/3D graphics operations).

- Memory mapped model – the network is viewed as a logical extension of paged memory, and paged-memory working set algorithms are used to maximize locality of data.

- Distributed object model – applications are structured as communicating objects. Objects are migrated by the network operating system to maximize throughput and minimize latency.

Some of the paradigms can be very traffic intensive. The compute cluster model can generate an average of almost a megabit per second of downstream bandwidth for display updates; the memory mapped model can utilize the entire available bandwidth of the Broadband-enabled virtual bus.

A recent set of initiatives called Grid computing **[7]** encompasses several of the models outlined above. The notable shift is that the network itself is becoming the interconnection bus of a massively parallel distributed virtual computer. As the bus

speed increases, and as the latency drops, applications are being developed that utilize the bus bandwidth and connectivity matrix.

P2P is not an aberration – it is an innate reflection of the speed and latency of always-on broadband networking, and is the first of many bandwidth-consumptive distributed applications that will be seen by MSOs.

What is needed are tools and mechanisms to classify and enforce traffic in a fair manner, both to the MSO provider and to the revenue generating subscriber, and which can quickly accommodate new distributed applications and application paradigms.

PACKET AND FLOW CLASSIFICATION

Packet Classification

The foundation of traffic enforcement is a function called Packet Classification. Packet Classification inspects incoming packets and hands off the packets to the QoS enforcement function (priority queueing, congestion management) of the packet processing engine[1].

Typically packet classification functions inspect source and destination addresses, port numbers, and priority fields. Some of the features that operators expect from packet classification include:

- Fair and policy-driven oversubscription management
- Monitoring and accounting
- Class of service management
- QoS on an application specific basis
- P2P awareness
- QoS on a subscriber or tiered service level basis.
- Usage based billing
- Denial of service protection

In order for packet classification services to be implemented, however, a set of filtering parameters – often called flows - must be learned and populated into tables used by the in-line packet classification function.

Methods for Learning Flows

There are two existing methods for learning flows – QoS-Smart Application Signaling and In-Line Flow Classification. HFC Cable standards currently are defining interfaces for the first method. This paper identifies requirements and interfaces for the second method. Both are ultimately required and both are compatible with DOCSIS in-line packet classification methods.

QoS-Smart Application Signaling

Judging by the existing technical work within HFC cable (DOCSIS 1.1 **[1]**, DOCSIS 2.0 **[2],** PacketCable **[3]**, and PacketCable DQOS **[4]**), there seems to be consensus that flow classification (learning flows) is a required element of the total QoS solution. The current solution focus is based on the paradigm of QoS-smart applications (smart with respect to QoS control). In this paradigm the application learns flows via some unspecified internal protocol mechanism. The application client and/or its trusted server then explicitly signals QoS control and authorization elements through standardized signaling interfaces to in-line networking nodes (CMTS edge router) or cascaded policy servers[2].

[1] "Packet classification" is used synonomously with "QoS enforcement" for the remainder of this paper.

[2] The PacketCable Multimedia project is currently underway and is defining an expanded QoS control model for multiple QoS-smart applications. The specifications are not yet public.

Figure 2 shows how the QoS-smart model is used within PacketCable[3] VOIP telephony. MGCP **[8]** and SDP **[9]** protocols communicate QoS control information at the application layer. The VOIP application server (Call Agent) uses PacketCable DQOS signaling to communicate flow classification parameters to the CMTS. The CMTS utilizes DOCSIS 1.1 to communicate flow classification parameters to the CM.

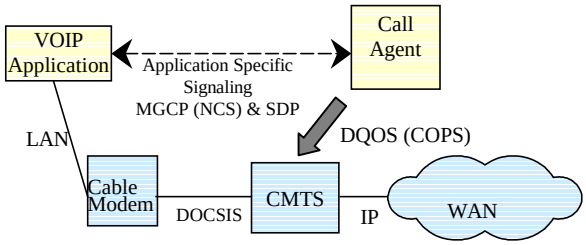

Figure 2 - PacketCable DQOS

The QoS-smart model works well for those applications that are developed in conformance to the model. Specific targeted services such as PacketCable VOIP telephony are beneficiaries of the QoS-smart style of learning flows.

However, for other new or legacy applications the QoS-smart model incurs long lead times between the specification, implementation, testing and deployment phases. Other delay inducing factors include:

(1) Today's applications development environment consists of many independent vested interests and non-collaborative standardization authorities. A major personal computer operating system vendor, for example, removed RSVP signaling and its associated APIs needed for application development from its latest generation windowing platform in 2002. Thus new applications on this platform have no formal QoS-smart signaling interface even available.

(2) In the multi-vendor application developer environment a variety of implementation platforms need to be harmonized – versions of Windows for clients, embedded systems with various RTOSs, Linux and Windows platforms for servers.

(3) Even given the presence of QoS signaling interfaces, the application developers must choose to utilize such interfaces.

From the subscriber's perspective new applications are appearing at an accelerating rate, and they are easy to access and install. The subscriber wishes to utilize QoS services for favorite applications (a revenue opportunity for operators), but since QoS-intelligence is lacking within the applications the subscriber (and any nearby neighbor sharing a common DOCSIS MAC domain) is destined for an unsatisfying experience.

In-Line Flow Classification

The second approach for flow classification (learning flows) is in-line classification. In this model all control-plane traffic is dynamically inspected and flows are dynamically learned.

The in-line classification engine is primed with external definitions of control-plane signaling mechanisms, applications, users, networks, and application signatures. The engine is also primed with policy definitions reflecting both operator and subscriber QoS policy attributes which govern applications traffic management in the packet classification phase.

[3] PacketCable Dynamic QoS (DQOS) is simplified for purpose of understanding. Other elements such as Record Keeping Server exist in the full architecture.

Figure 3 shows the in-line flow classification model. All packets received from any port are inspected by the Flow Classification function (FC) which tracks and maintains current state for the control flows of active application instances. The Flow Classification function creates parameters in a flow description table. Typically the parameters consist of IP addresses, UDP port numbers, protocol Ids and various traffic handling policies for the flow.

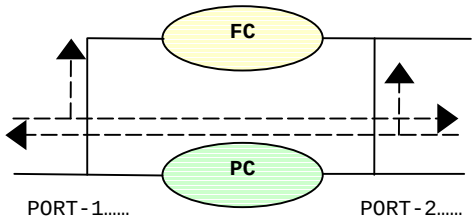

Figure 3 - In-line flow & packet classification

The Packet Classification (PC) function inspects all packets in both directions and categorizes each packet into a specific flow based upon the parameters stored in the flow description table. The packets are placed into queues and traffic policies are implemented according to the flow definitions in the flow description table (priority, rate-limit, packet discard).

This model is similar to the model for managing virus signatures in virus checking systems today. The difference is that the classification functions and classification definitions are maintained within the network, rather than on the end-subscriber's PC or workstation.

The advantage of the in-line classification method is that new applications can be rapidly identified and given packet classification and QoS enforcement functionality. Rapid distribution of new classification parameters and subsequent configuration by operators and subscribers quickly implements policies for new applications. A reasonable goal is to reduce the time for supplying QoS functionality (and generate revenue) for new applications from months (or years) to weeks (or days or hours).

Flow Classification Mechanisms

Flow classification[4] requires a number of processing-intensive mechanisms:

1. Portless flows – the flow is simply defined as a combination of IP addresses, and perhaps also a protocol identifier. No real learning is required other than detection of active packet flow with timeouts.

2. Fixed port mapping flows – the flow is simply defined as IP addresses and port numbers. No real learning is required other than detection of active packet flow to/from fixe addresses/ports with timeouts for inactivity.

3. TCP with well-known-ports – TCP has a standard application specific method for establishing flows [ref: IANA assigned number authority]. Inspection of TCP packets and looking for the session establishment commands (TCP SYN packets) can identify specific flows. The well known port number identifies the application.

4. Out of band control protocols – many application protocols use out of band methods to communicate IP addresses and port numbers. All

[4] The term "stateful classification" is sometimes used and is equivalent to flow classification as used in this paper.

packets of the out of band control protocol are monitored, and IP addresses and port numbers are extracted from the control signaling. H.323, MGCP, and SIP are three examples of out of band control protocols.

5. Protocols that use random port numbers for session setup. Many applications such as some P2P file transfer applications use random not-well-known port numbers. These are sometimes called "port hopping" applications. Applications signatures must be detected for these protocols (see below).

6. Masquerading protocols – applications masquerade as existing well-known applications protocols such as HTTP (web) and FTP (file transfer). In some cases these are well-intentioned methods to pass through firewalls without having to impose on firewall managers. For example, some implementations of voice/video instant messenger services can be carried as HTTP traffic both to utilize HTTP security and to achieve firewall traversal. In other cases they are pernicious ways of fooling the network into thinking this is a friendly application (e.g. KaZaA), and obtain preferential bandwidth and access rights. Application signatures must be utilized to ferret out any of these sorts of flows.

For several of the application types above, application signatures are required. Application signatures are defined as application-specific protocol elements embedded deep within packets, e.g. HTTP fields, which contain application specific values. There is no standard for definition of application signatures, and in some cases multiple fields and Boolean expressions must be computed before a specific application signature is matched.

An in-line flow classification function must inspect all combinations of the above application types in control streams in order to unambiguously differentiate application flows. Once the flow is identified, then specific flows and policies can be defined for use by the in-line packet classification function.

NEW TOOLS

The in-line flow classifier is a new architectural element. Since it inspects all packets in order to classify flows, it also performs packet classification functions in order to enforce administrator and subscriber defined policies. We call this element the Classification and Enforcement Engine (C&EE). See **[10, 11]** for a specific example of a C&EE and its application in managing peer to peer traffic.

The C&EE is typically external to existing network nodes (e.g. CMTS and CM), although it could be implemented internal to a CMTS or CM if the platform has enough packet processing horsepower and is not limited by inflexible ASIC functionality.

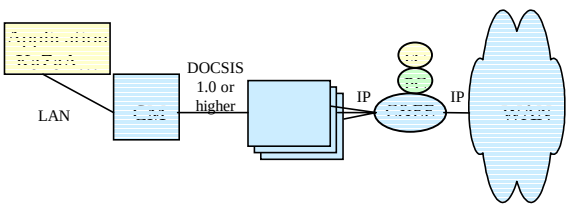

Figure 4 - C&EE with CMTS/CM

The C&EE provides both flow classification (FC) and packet classification

(PC) functions. Policies can be defined and enforced for all applications, including P2P file transfer, even in DOCSIS 1.0 systems. New applications can be detected and configured by the system administrator. Database updates from the C&EE supplier can update the knowledge base of application and protocol types with quick turnaround times.

End-to-End System Architecture Evolution

Since the C&EE, CMTS, and CM are all capable of providing packet classification (PC) functions (both DOCSIS 1.1 and 2.0), a future opportunity exists to integrate all networking elements into a strong end-to-end QoS management architecture. In the integrated architecture the flow classification and packet classification function of the C&EE is combined with the packet classification functions of the CMTS and CM via standard signaling interfaces. Administrator and Subscriber defined traffic policies can then be concurrently applied to both QoS-smart applications and QoS-unaware applications.

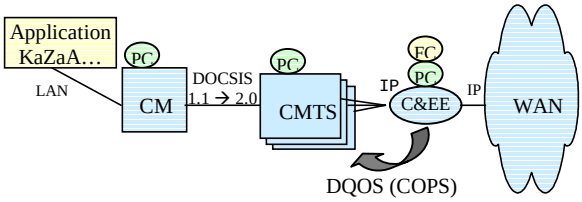

Figure 5 - C&EE and DQOS

Figure 5 shows a configuration for the integrated end-to-end system. The C&EE aggregates traffic from multiple CMTS platforms (and their downstream CMs and applications). The C&EE performs flow classification (FC) and utilizes the DQOS signaling interface to communicate relevant parameters and policies to the packet classification (PC) function of the CMTS. The CMTS utilizes the DOCSIS 1.1 (and above) signaling interface to communicate relevant parameters to the packet classification function of the CM. No application server is required in order to support or add new applications. This integration is achieved using existing PacketCable DQOS signaling interfaces and parameter definitions.

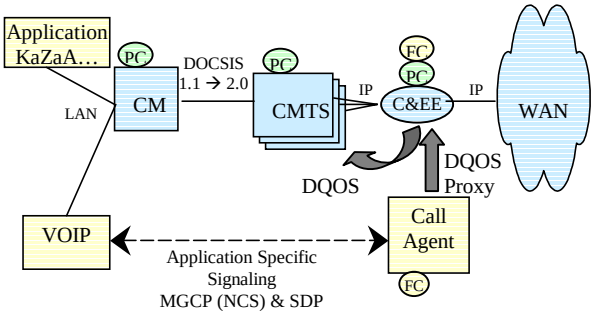

Figure 6 – C&EE with QoS-smart Server

Figure 6 shows concurrent operation of C&EE in-line flow classification with QoS-Smart application servers flow classification (e.g. PacketCable Call Agent). In this configuration the C&EE node proxies between the downstream CMTS nodes and the Policy Server (e.g. PacketCable Telephony Call Agent). In this configuration both QoS-smart flow learning (e.g. for PacketCable Telephony) and in-line flow learning can coexist.

Some protocol and parameter extensions are likely required in order to maximize the functionality of the full end-to-end QoS architecture[5]. For example, the total available bandwidth is divided between the flows known by the QoS smart application (PacketCable Telephony) and the C&EE node. Also, given the wider variety of

[5] The current proposal is based upon the PacketCable DQOS framework. Once PacketCable Multimedia is published appropriate modifications can be introduced to the proposal.

application types supported, some new traffic policy types may need to be defined.

CONCLUSION

A new approach is proposed for dealing with the explosive growth of new applications, new application types, and rising subscriber expectations for fair and configurable quality of service in DOCSIS HFC networks. The approach features a new element which can be incrementally added to the end-to-end architecture. This element is called the Classification and Enforcement Engine (C&EE) and provides in-line flow classification functions for both existing and new types of application traffic. The C&EE is compatible with the current QoS-Smart model for flow learning in PacketCable, DOCSIS, and emerging PacketCable Multimedia standards.

The C&EE can be implemented in DOCSIS 1.0 systems, and, with appropriate future extensions, can utilize the PacketCable DQOS signaling interface to fully utilize the packet classification functions contained within the CMTS and CM for DOCSIS 1.1 (and higher) systems.

Using new tools like the C&EE the operators and subscribers will have the ability to manage and control bandwidth utilization on an application by application basis. More importantly, operators will both retain subscribers and reap new revenue for managed bandwidth services.

REFERENCES

[1] CableLabs, DOCSIS™ 1.1, "Data-Over-Cable Service Interface Specifications, Radio Frequency Interface Specification - SP-RFIv1.1-I09-020830", 8/30/2002.

[2] CableLabs, DOCSIS™ 2.0, "Data-Over-Cable Service Interface Specifications, Radio Frequency Interface Specification SP-RFIv2.0-I03-021218", 12/18/2002.

[3] CableLabs, **"**PacketCable™ 1.0 Architecture Framework Technical Report - PKT-TR-ARCH-V01-991201", 12/1/1999.

[4] CableLabs, "PacketCable™ Dynamic Quality-of-Service Specification - PKT-SP-DQOS-I05-021127", 11/27/2002.

[5] Marshall McLuhan - The Medium is the Massage. New York: Bantam Books, 1967.

[6] Pickens, "Transport protocol revolution", in SNA and TCP/IP Enterprise Networking, Manning Publications, 1998.

[7] Ian Foster and Adriana Iamnitchi, "On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing", http://iptps03.cs.berkeley.edu/final-papers/death_taxes.pdf.

[8] CableLabs, "PacketCable™ Network-Based Call Signaling Protocol Specification - PKT-SP-EC-MGCP-I06-021127", 11/27/2002.

[9] IETF, RFC 2327 "SDP: Session Description Protocol", April 1998.

[10] Allot Communications Inc., "NetEnforcer Data Sheet", Allot Communciations, Inc., www.allot.com.

[11] Allot Communications, Inc., "KaZaA v2 and other new P2P Applications Technical Note", www.allot.com.

John Pickens is a Technical Advisor to Allot Communications, Inc. and can be contacted at jpickens@ieee.org.

Greg Hutterer is the Director of Service Provider Sales at Allot Communications, Inc. and can be contacted at ghutterer@allot.com or (952) 944-3100.