Submission Reference Number: SIM-S06-107

Title:
A Simulation Model of the DOCSIS Protocol

Authors:
Jim Martin and James Westall
School of Computing
Clemson University
{jim.martin,westall}@cs.clemson.edu

Contact Author Information:
Jim Martin
211 McAdams
Clemson University
Clemson, South Carolina 29634
Email: jim.martin@cs.clemson.edu
Phone: 864 656 4529
Fax: 864 656 0145
A Simulation Model of the DOCSIS Protocol

Jim Martin and James Westall
School of Computing
Clemson University
{jim.martin,westall}@cs.clemson.edu

Abstract
The number of households and businesses using HFC cable networks for Internet access is rapidly approaching 40 million in the United States. The cable industry has standardized on a single medium access control (MAC) and physical layer standard, the Data Over Cable Service Interface Specification (DOCSIS). The MAC layer of the emerging IEEE 802.16 broadband wireless access standard is also based upon DOCSIS. Thus, the performance of DOCSIS is now and will remain a critical element in the overall performance of shared medium broadband access networks. Despite this fact, public domain tools that may be used to assess and improve the performance of DOCSIS have been slow to emerge. To address this problem, we have implemented a simulation model of the DOCSIS MAC layer for the 'ns' network simulation tool.

Due to the complexity of DOCSIS and because the specifications are purposely incomplete, developing an accurate simulation is quite challenging. In this paper we present analytic and live network evidence that the simulation accurately reflects the behavior of a DOCSIS network under a limited set of workloads.

Keywords: Simulation methods, broadband access, HFC cable networks, DOCSIS, TCP performance

1. Introduction
Early cable broadband access networks employed proprietary protocols. Performance was characterized by highly asymmetric channel capacities and high loss rates due to poor channel signal-to-noise properties[1,2]. The deployment of hybrid fiber coaxial (HFC) systems pushed fiber closer to the endpoints and greatly improved performance. In the late 1990’s the cable industry developed a set of standards, collectively referred to as Data-Over-Cable Service Interface Specification (DOCSIS), for transmitting data over HFC cable networks [3,4]. The DOCSIS standards define the physical and MAC layers and also address security, operations system support (OSS), equipment interfaces, and equipment validation. Standard protocols, better performing equipment, and competitive pricing have fueled the steady growth of broadband cable.

According to a Pew survey, nearly 65% of adults in the United States access the Internet at least once each day [5]. The study also indicates that there are 66 million households equipped with broadband access, and that approximately 54% of these households use cable. Advancements in the technology driving HFC cable networks continue. Sophisticated modulation techniques augmented with channel bonding will increase data rates from the current tens of Mbps to hundreds of Mbps in both the upstream and downstream directions.

When compared to other emerging network technologies, (e.g. ad-hoc and sensor networks), there is little published research on the subject of modern cable networks. There are two main reasons for this. First, due to complexity and cost, there are no open source DOCSIS platforms that are available to researchers. Second, in contrast to the Internet community where academic researchers can introduce new protocols or protocol enhancements through the IETF’s RFC process, the HFC cable and Worldwide Interoperability for Microwave Access (WiMAX) standards are developed in members-only industry consortiums. The net result is that the evolution of DOCSIS is being directed by industry with little involvement of academia. As a first step in enabling the academic community to participate in the evaluation and evolution of DOCSIS, we have developed a simulation model of the DOCSIS MAC protocol for the popular 'ns' network simulator tool [6].

The challenges of performing simulation-based studies of complex systems are well known. It has been argued that research based solely on simulation lacks credibility [7]. Recent publications note that wireless research is particularly sensitive to physical layer assumptions [8-11]. The authors of [9] indicate that the choice of appropriate experimental scenarios is just as crucial as the use of valid channel models.

We encountered significant challenges in developing the DOCSIS simulation. The behavior of the system is quite sensitive to parameter selection, and the specification

---

1 The ‘ns’ DOCSIS simulation model is available at http://people.clemson.edu/~jmarty/docsis.html.
leaves significant room for implementation decisions, leaving key parameter selection decisions to the cable system operator. Thus the number of “correct” realizations of DOCSIS is nearly limitless, and the process of “validating” a simulator for all of them unending.

These issues motivated the work reported in this paper. While our overall research agenda is to participate in the continued evolution of DOCSIS to better support current and next generation shared medium access networks, our objectives in this paper are limited and two-fold. The first is to describe those operational elements of the DOCSIS protocols that are modeled in our simulation. The second is to demonstrate that the performance of a set of selected workloads on the simulated system is consistent with the performance observed on operational HFC networks. Aspects of this research are applicable to 802.16 (WiMAX) networks as its channel allocation mechanism is also DOCSIS-based. Differences between the HFC physical layer and WiMAX physical layer prevent results from being directly compared. Nevertheless, the methodology we describe to validate a wired DOCSIS network model can be applied for a WiMAX network model. In future work, we intend to extend our simulation model for WiMAX. The focus of this paper, however, is on HFC networks.

The remainder of this paper is organized as follows. In section 2 we describe essential elements of the DOCSIS MAC protocol. Section 3 surveys related work. The design of the simulator and those elements of DOCSIS that it supports are discussed in section 4. Our approach to validation and insights gained in the validation process follow in section 5, and we conclude is section 6.

2. Background

Early packet radio networks using MAC protocols such as slotted Aloha operated well at low loads but did not scale with increased load [12,13]. Reservation schemes and hybrid schemes based on a random access reservation mechanisms and TDMA were shown to provide improved performance at high loads. The DOCSIS protocol is a derivation of the Aloha Reservation protocol which managed bandwidth in a distributed manner using the slotted Aloha request mechanism [14]. Frames are divided into equal length slots, one of which is designated as a “reservation slot”. The reservation slot is further divided into smaller mini-slots. A station with data to send broadcasts a request during a randomly selected reservation mini-slot. If the reservation is successful, the station learns the slot(s) it was allocated and transmits its data using the allocated slots. Subsequent cellular-based packet radio networks as well as many of the wireless Asynchronous Transfer Mode (ATM) proposals were based on variations of the Packet Reservation Multiple Access (PRMA) protocol [12,15,16]. To support hundreds of data users sharing a channel in a cable network, the IEEE 802.14 working group used a centralized variant of the Aloha Reservation protocol as the MAC protocol that extended ATM services over a cable channel. This work eventually led to DOCSIS.

A typical DOCSIS network is shown in Figure 1. The cable system operator hosts the cable modem termination system (CMTS) units that interact with cable modems (CMs) deployed at subscriber locations. A modern CMTS houses multiple ‘blades’ with each blade supporting one or more HFC domains (one downstream channel with four upstream channels). Six MHz (or greater) bandwidth is allocated from the 88-860 MHz spectrum for each downstream channel, and upstream channels are allocated from the 5 – 52 MHz frequency range.

In the downstream direction, a single sender (the CMTS) transmits to the attached CMs. Downstream data rates presently vary from 10 Mbps to 50 Mbps. IP packets sent downstream are divided into 188-byte MPEG frames. Each CM has a unique MAC address and will receive only frames that are addressed to its MAC address or to the broadcast address. Computers owned by the subscriber connect to the CM through an Ethernet or USB interface.

In the upstream direction multiple senders (CMs) share a channel that supports data rates in the range of 5 Mbps to 10 Mbps. The upstream transmission model is shared access using time division multiple access (TDMA) with a random access contention-based reservation mechanism. IP packets that are sent upstream are encapsulated in a DOCSIS frame and transmitted during assigned slots. If a packet does not fit into the number of contiguous slots that were allocated it is fragmented into multiple DOCSIS frames.

![Diagram of Typical DOCSIS Network](image)

Figure 1. Typical DOCSIS network

Traffic is classified by service flow. Each service flow is assigned a unique numeric identifier commonly referred to as the SID. For example, a configuration that supports
telephony and best effort data typically has four service flows: one each for the upstream and downstream VoIP traffic and one each for the upstream and downstream best effort traffic. DOCSIS maps service flows to one of several ATM-like services including best effort, unsolicited grant service (UGS), which is equivalent to ATM’s constant bit rate service, and real-time polling (rtPS), which is similar to ATM’s variable bit rate service. As in ATM, different performance guarantees are available for each service class. In the UGS service the CMTS periodically provides unsolicited grants of upstream slots to the CM, in the rtPS service the CMTS periodically asks the CM if it needs bandwidth, and in best effort service bandwidth is allocated on-demand using a contention-based request mechanism.

The upstream channel is time-division multiplexed using fixed size transmission slots. These slots are referred to as mini-slots in the DOCSIS specification, but we use the term slot to in this paper. The capacity of a slot is fixed on a given DOCSIS network and is typically on the order of 8 to 16 bytes. Permission to transmit data in a block of one or more adjacent slots must be granted to a CM by the CMTS. The CMTS provides such grants by broadcasting on the upstream channel a control packet known as the Upstream Channel Allocation Map. This packet is commonly referred to simply as the MAP. The number of slots in a frame that is described by a single MAP can vary but is typically fixed in a given DOCSIS network to a time domain value between one and ten milliseconds. This value is known as the MAP time. MAP messages must describe slots far enough in the future that CMs have sufficient time to prepare for upstream transmission allocations. In most implementations the MAP time is constant, but dynamically varying MAP times are permitted in the specification.

Information elements carried in the MAP describe the usage of the slots comprising next upstream frame. These include: data slots granted to specific CMs; management related slots; request slots reserved for individual rtPS service flows; and contention slots in which CMs may request data grants for best effort upstream traffic.

When a CM makes a contention request, it selects a random number within a backoff window. The initial window size is determined by a starting backoff range value and a maximum backoff range value that is specified by the CMTS and maintained by the CM. The random number is selected using a uniform distribution in the range of \([1, \text{backoff window}]\). The selected number represents the number of contention request slots the CM must let pass before transmitting a request. Depending upon load, it is possible for a CM to have to delay multiple MAP times before transmitting its request. After a CM transmits the request, if the next MAP does not contain a grant or a grant pending from the CMTS, the CM assumes a collision has occurred and doubles the size of the backoff window. The contention request cycle continues until a grant or grant pending is received or 16 consecutive failures have occurred in which case the packet is dropped. The backoff window is limited by the configured maximum backoff range parameter sent to all CMs. A CM may request bandwidth sufficient to transport multiple IP packets in a single DOCSIS frame by issuing a concatenated bandwidth request. To further reduce contention, a CM is permitted to piggyback a request for bandwidth on an upstream data frame, if it has another IP packet in its output queue when it begins to transmit the current packet.

The DOCSIS specification does not provide guidance in a number of aspects that significantly impact performance. There are many situations in which the use of either piggybacking or concatenation can significantly improve performance. Since there are times in which a CM can do either piggybacking or concatenation, the specific behavior is implementation dependent. All aspects of scheduling at the CMTS are left unspecified. The only constraint is that service specific performance measures are upheld. For UGS and rtPS service flows, unsolicited grants and periodic polls must be respectively provided within the tolerated jitter specification.

3. Related Work

In [1], the authors found that TCP throughput over an 802.14-based HFC network is low primarily due to ACK compression. While assumptions made by the authors (such as high loss rates in the upstream path) are no longer true, our recent results do confirm that DOCSIS induces ACK compression. Other early studies looked at TCP enhancements when operating over HFC networks suffering from high channel error rates or high collision rates due to heavy workloads [2,17,18].

More recent analysis of DOCSIS has been performed using either analytic models or simulation models [20-23]. The work in [20] was based on the Opnet DOCSIS model [19]. The work in [21] explores system delays based on a Markov model of DOCSIS. The work in [22] was based on an in-house simulation model. Due to differences in methods and implementations, it is not possible to compare published results with our results. In [23], the author studied DOCSIS primarily with analytic methods although an ‘ns’ simulation model was also developed to validate the analytic results. However, the fidelity of the model was sufficient only to confirm that DOCSIS impacts TCP performance primarily because the upstream channel is packet rate limited. No prior work of which we are aware includes model verification and validation using live DOCSIS systems.
The prior work does not offer a unified, clear understanding of how DOCSIS behaves. In contrast, 802.11 technology has been in use approximately as long as DOCSIS and is considered ‘well understood’ [24,25]. Barriers such as protocol complexity and cost of equipment have hindered the study of DOCSIS. Our work provides a framework that includes analysis, tools, and techniques that collectively will facilitate future research in DOCSIS.

4. The Simulation Model

The simulation model implements the DOCSIS architecture defined in [3] with the following restrictions: 1) CMs are limited to a single default best effort service flow and a single UGS or rtPS service flow; 2) the model is limited to one upstream channel and one downstream channel; 3) the model does not support dynamic service provisioning; 4) physical layer impairments are not modeled; 5) the model assumes that the CMTS and the CM clocks are synchronized.

To account for forward error correction (FEC) overhead in the physical layer, we reduce the upstream channel capacity by 8% and by 4.9% in the downstream channel. This approximation was suggested by Cisco [26]. The contention backoff range that is sent to each CM in downstream MAP messages is statically set by a configuration parameter. Both concatenation and piggybacking can be enabled or disabled. If concatenation is enabled, a configuration parameter specifies the maximum number of IP packets (or bytes) that can comprise a single concatenated transmission. After a CM transmits a frame, the CM waits until the next MAP to arrive. At this time, if the CM has more than one packet waiting for upstream transmission, it must decide to use either piggybacking or concatenation. Our implementation will use concatenation rather than piggybacking.

To account for forward error correction (FEC) overhead, we reduce the upstream channel capacity by 8% and by 4.9% in the downstream channel. This approximation was suggested by Cisco [26]. The contention backoff range that is sent to each CM in downstream MAP messages is statically set by a configuration parameter. Both concatenation and piggybacking can be enabled or disabled. If concatenation is enabled, a configuration parameter specifies the maximum number of IP packets (or bytes) that can comprise a single concatenated transmission. After a CM transmits a frame, the CM waits until the next MAP to arrive. At this time, if the CM has more than one packet waiting for upstream transmission, it must decide to use either piggybacking or concatenation. Our implementation will use concatenation rather than piggybacking.

The bandwidth scheduler operating at the simulated CMTS executes every MAP time. The scheduler examines requests for upstream bandwidth and assigns slots in the next MAP time using a hybrid earliest deadline first scheduling policy[27]. As illustrated in Figure 2, the scheduler maintains four queues: two for periodic bandwidth allocations; and two for on-demand requests for bandwidth. Periodic grants from either UGS or rtPS service flows are maintained in separate queues, but they are effectively treated as a single queue of bandwidth requests arranged in earliest deadline first order. The elements in the queues are ‘jobs’ with deadlines calculated as follows:

- UGS deadline: arrival time\(^2\) + nominal grant interval
- rtPS poll deadline: arrival time + nominal polling interval
- On-demand deadline: arrival time + max tolerated access delay

UGS grants must be allocated within the jitter tolerance associated with the service. Likewise, the specified jitter tolerance associated with rtPS polls must be met. These periodic requests have strict priority over the two best effort queues. Periodic requests that miss their deadlines are deleted and a ‘provisioning error’ statistic is incremented.

After periodic requests having deadlines that occur during the current MAP time are allocated, management slots and the required minimum number of contention slots are allocated. The remaining slots are then allocated to the two classes of on-demand requests. When a CM requests bandwidth through a response to an rtPS poll, the request is inserted in the rtPS aperiodic queue. The scheduler divides the number of available slots by a weight factor allowing the system to treat rtPS data requests with a different priority level than a data request that arrives through either contention or piggybacked request mechanisms. The maximum tolerated access delay allows the CMTS to control the level of backlogged requests. We set this to 1 second. If an aperiodic job is not processed by its deadline, the job is deleted and a ‘denied request’ statistic is incremented.

Figure 2 illustrates that the scheduling process includes a packing operation in which allocations are arranged within the MAP. The simulation uses a form of next fit bin packing wherein jobs are inserted in the MAP in a manner that meets periodic jitter requirements and that optimizes performance. By default, any unused slots are allocated for additional contention requests.

As noted earlier the DOCSIS specification permits the MAP time to grow up to a configured number of slots (known as the MAP lookahead). Our simulator can be configured to support this, but for all of the results reported here the MAP time was fixed.

2 The arrival time is the time at which the request for bandwidth arrives at the scheduler. For periodic traffic, the scheduler artificially creates an arrival at fixed intervals.
The scheduling algorithm defines system performance. The algorithm that we have implemented, which we refer to as the “DOCSIS compliant” scheduler, serves as a baseline. In the results reported in this paper, we focus on relatively simple scenarios that do not involve UGS or rtPS services. Therefore the generic ‘DOCSIS compliant’ scheduler that we have implemented should match the behavior of an actual system in the scenarios that are studied.

Figure 3. Upstream transmission scenario with variable MAP times

Figure 3 illustrates the upstream transmission of a 1500 byte IP datagram from a TCP source to a sink located outside the HFC network. Time progresses in the downward direction. The small dark square box positioned at the beginning of each MAP time represents the transmission of the MAP message in the downstream direction. The simulated CMTS sends a MAP message at the beginning of each MAP time.

An IP packet arrives at the CM during the j-th MAP at time T-0. The CM sends a bandwidth request message at time T-1 (in a contention request slot) and receives the data grant at time T-2. The grant is located in the third MAP time. The CM begins to transmit the frame at time T-3 and the last bit of the frame is received by the CMTS at time T-4. The time between T-3 and T-0 is the access delay which represents the total time a packet is delayed over the DOCSIS network not including transmission and propagation time experienced by the data packet (we refer to this delay as \( t_{\text{request}} \)). In our implementation, the \( t_{\text{request}} \) experienced by a CM in an unloaded system will be two or three MAP times depending on when the packet arrives at the CM with respect to MAP time intervals.

The illustration presumes that the number of slots required to transmit the entire packet can be granted within a single MAP. If this is not the case, then the CMTS will issue a sequence of partial grants, requiring the CM to send DOCSIS (not IP) fragments. Reassembly of the full IP packet is performed by the CMTS.

Packets that arrive at a CM from the subscriber interface are queued by the CM until they can be transmitted. The size of the upstream CM queue is a configuration parameter.

In prior work we showed that performance can deteriorate as the MAP time grows [28,29]. The contention request allocation strategy and the use of fragmentation, concatenation and piggybacking also have significant impact on system performance.

5. Model Validation

As noted in the introduction, it can be argued that a simulation of a system as complex as a DOCSIS network can never be claimed to have been truly verified or validated, and we certainly make no claims to having done so here. Nevertheless, it is important to establish, insofar as it is possible, that the results produced by the simulation are not inconsistent with behavior that can be observed on an operational DOCSIS network.

To that end, we have employed a three phased approach in which carefully selected workloads were simulated, and the results were evaluated with respect to increasingly complex sources of ground truth. The approach is illustrated in Figure 4. The simplest but most limiting source of ground truth is the analytic model. While it can almost instantly provide performance measures for a wide range of parameters, all but the simplest of workloads are analytically intractable. Our analytic model assumes only a single, always-on UDP-like traffic source that maintains a packet backlog at the CM. Nevertheless, it proved useful in the verification of the simulation program and it provides a useful first step in the validation process.

Figure 4. Validation approach

To gain confidence that the DOCSIS simulation model produced correct behavior under somewhat more complex (and thus analytically intractable) workloads, we measured
performance of selected benchmark workloads on two operational DOCSIS networks: a private laboratory network that we operate; and a public network operated by Charter Communications.

Criteria used in selecting benchmark workloads included the following. Workloads should be simple and provide repeatable behavior to facilitate identification of the source(s) of anomalous behavior in the simulation. Some workloads should be explicitly designed to elicit known idiosyncratic performance characteristics of the DOCSIS protocol.

Therefore, in this first phase of validation, none of the workloads introduces explicitly stochastic behavior such as would be caused by the use of pseudo-random inter-packet times. Furthermore, even implicitly stochastic behavior (as would be caused by multiple concurrent upstream traffic sources) is also not present. In the study involving the DOCSIS network of Charter Communications stochastic behavior obviously is present, but is not introduced by our benchmark workloads.

Stochastic elements are present in benchmark workloads in tests that are ongoing. Nevertheless, we remain convinced that, for a simulation of a system as complex as a DOCSIS network, a very disciplined and incremental approach to validation is the most productive.

5.1 Analytic Validation

The upstream behavior of DOCSIS is similar to slotted Aloha with reservations [30-32]. Following the method presented in [33], we define the maximum upstream application throughput, \( T_{\text{max,up}} \), to be

\[
T_{\text{max,up}} = \frac{D_{\text{perCycle}}}{t_{\text{data}} + t_{\text{request}}}
\]

where \( D_{\text{perCycle}} \) is the amount of user data sent upstream in one reservation request cycle, \( t_{\text{data}} \) is the upstream transmission and propagation time of the data, and \( t_{\text{request}} \) is the delay associated with the request process. In this section, we consider an analytic model that computes the upstream throughput that could be obtained by a single application flow. The model does not consider loss or collision that might be caused by competing traffic. While the single-flow assumption is clearly limiting, it allows us to compare basic operation of the MAC analytic and simulation models to live DOCSIS networks under controlled conditions. For brevity, we only consider the backlogged case where a single CM always has 1500 byte IP packets to send. Figure 3 illustrates the \( t_{\text{request}} \) and the \( t_{\text{data}} \) delays in an upstream operation. The \( t_{\text{request}} \) represents the total delay experienced by the packet from its arrival at the CM until its first bit is transmitted upstream. The \( t_{\text{data}} \) is the transmission and the propagation time of the upstream data frame.

For brevity, we also omit the details of the analytic model and focus upon its use in validating the simulation. Appendix A shows the derivation of the \( t_{\text{request}} \) for the case of no piggybacking and concatenation and a fixed number of contention slots in each MAP. Refer to [34] for a complete description of the analytic model. The model incorporates basic configuration information such as the number of slots in a MAP, the number of management slots and the number of contention request slots per MAP, the backoff range, and the stochastic nature of the backoff process. The model optionally accounts for piggybacking and concatenation. In this section we consider results from the analytic model with piggybacking and concatenation disabled.

The analytic model presumes a single always-on source in which a packet backlog is always present in the CM. Thus, it also closely models the behavior of a sustained TCP connection in which (1) there is no packet loss and (2) TCP window size and the rate of ACK arrivals is such that the assumptions related to the queuing behavior at the CM are satisfied.

5.1.1 Simulation Methodology

The simulated network used in the analytic evaluation is illustrated in Figure 5. One FTP flow is active between CM-1 and the server S-1. The maximum TCP window configuration setting was 11 packets which was sufficient to keep the upstream transmission queue at CM-1 always filled with data but to avoid queue overflow. We disabled piggybacking and concatenation in the simulation model so that it is consistent with the assumptions of the analytic model.

![Simulation network model and parameters](image-url)
We performed multiple runs varying the MAP time parameter from a value of 0.001 seconds to a maximum value of 0.012 seconds. Each data point of the simulation model curve is the average TCP throughput observed in a single simulation that ran for 1000 simulated seconds. We did not use multiple runs as the variation over multiple runs was very small, less than 0.5%. To test the level of certainty associated with the statistic, we ran the MAP time of 0.002 seconds point illustrated in Figure 6b for 10 iterations. The 99% confidence interval was within 0.23% about the mean.

Figures 6a and 6b show TCP throughput in Mbps predicted by the analytic model and throughput observed in the simulation experiment. A backoff range of eight slots was used to obtain the results shown in Figure 6a. This parameter was increased to 64 slots for the results shown in Figure 6b. The throughput obtained by the simulation closely tracks the throughput predicted by the analytic model in both cases with the maximum difference never exceeding 10%. Two results are evident. First, upstream TCP throughput deteriorates as the MAP time increases because a larger MAP time increases the $t_{\text{request}}$ delay. Second, the results show that increasing the backoff range also reduces throughput because a higher backoff range also increases the $t_{\text{request}}$ delay.

5.2 Validation with Operational Networks

In this section we analyze the performance of somewhat less constrained workloads when run on the simulator and on operational DOCSIS networks. The first studies presented were conducted on a private DOCSIS testbed located in our research lab. The second employed a lab computer located at Clemson University and a home computer connected to the Internet via Charter Communication’s public DOCSIS network.

5.2.1 Network 1: The DOCSIS testbed

The testbed consists of four CMs (two Cisco uBR905s and two Motorola SurfBoards), a Cisco uBR7110 CMTS, five Linux-based PCs, and a simple RF plant. The downstream service rates on the DOCSIS network were set to 10 Mbps and the upstream service rates were 1 Mbps. The CMTS is connected to a private 100 Mbps Ethernet which hosts the client system. The testbed is particularly useful as it provides a controlled environment in which the performance of benchmark workloads is never affected by competing traffic.

5.2.1.1 The UDP Echo Workload

The first testbed study involved a UDP echo application between a Linux “server” connected to one of the CMs and a Linux “client” connected to the Fast Ethernet network upstream of the CMTS. The client sent a periodic stream of small (64 byte) UDP packets downstream to the server which echoed each packet back to the client. We obtained tcpdump traces at both the client and the server [35]. This study was designed to elicit idiosyncratic characteristics of DOCSIS performance related to the length of the MAP time. For the results shown in figures 7 and 8, both the base inter-packet departure time and the MAP time were 0.002 seconds.

Figures 7a through 7d show the empirical distributions of the inter-packet departure and arrival times in the form of probability density histograms using a bucket size of 200 microseconds. Figure 7a shows that more than 99% of the samples depart on time. Figure 7b illustrates that processing delays at the CMTS and the CM introduce minimal increase in jitter as the packets traveled...
downstream. Figure 7c shows that server processing introduces minimal but measurable additional jitter. Interdeparture times range from 0.0015 seconds to 0.0025 seconds. The effect of DOCSIS scheduling finally appears in figure 7d where 55% of the packets have an interarrival time of less than 200 microseconds. This indicates that about 55% of packets transmitted by the server arrived at the CM when there was at least one packet already waiting. The CM sends these packets in a single, concatenated transmission.

Figure 7. DOCSIS testbed observed departure and arrival distributions (2 ms packet spacing)

Figure 8. Simulation departure and arrival distributions (2 ms packet spacing)
Figure 9. DOCSIS testbed observed departure and arrival distributions (5 ms packet spacing)

Figure 10. Simulation departure and arrival distributions (5 ms packet spacing)
The remaining 45% of the UDP packets arrived at the CM when no other packets were queued. Therefore, these packets suffer a lengthy delay of two MAP times plus a small amount of delay that depends on the accumulated jitter and scheduling delays over the network. Based on the modes in the 0.004 through 0.005 second range of Figure 7d, one would conjecture that the MAP time is 0.002 seconds and, on our DOCSIS testbed, this was indeed the case.

Figure 8 illustrates the results of a simulation of the upstream portion of the testbed experiment illustrated in Figure 7. The simulation was configured as illustrated in Figure 5 except that only one CM is active. The source attached to this CM transmits a 64 byte packet every 0.002 seconds. A sink is connected to the CMTS with a 100 Mbps link. Figure 8a shows the inter-packet departure times at the source. Figure 8b illustrates the inter-packet arrivals at the sink. Figures 8a and 8b are comparable to Figures 7c and 7d. As in the testbed, approximately 50% of the packets are sent upstream in concatenated frames. Unlike the testbed results, the remaining 50% of the density falls within one mode rather than three. Additional experimentation tells us that the two small modes on each side of the large mode at 0.002 seconds observed in Figures 7b and 7c do not induce the three modes observed in Figure 7d. We conjecture that the Cisco CMTS ‘packs’ the MAP differently than the simulation. For example the Cisco CMTS might distribute contention request slots in groups spread over the MAP causing data transmission start times to vary within the MAP. For the static conditions associated with the simulation, upstream transmissions always occur at approximately the same time in a MAP.

To investigate the impact of decoupling the inter-packet departure time from the MAP time, the UDP echo experiment was repeated with the inter-packet departure times set to 0.005 seconds. The results are shown in figures 9 and 10. Figure 9d shows two modes at 0.0045 and 0.006 seconds. We rely on the simulation model to help explain this result. Figure 10a shows the inter-packet departure times at the sending node. Figure 10b illustrates the inter-packet arrivals at the sink. When packets arrive at the CM for upstream delivery every 0.005 seconds, the timing is such that approximately 45% of packets arrive while the CM is waiting for the grant from the previous packet which allows the CM to issue a piggybacked request for bandwidth when it transmits the previous packet. In our implementation, the access delay associated with a piggyback request is 2 MAP times which explains the mode at 0.004 seconds. The packets that arrive when the CM has already transmitted the prior packet are subject to a lengthy delay of 3 MAP times.

The results illustrated in Figure 9d suggest that the DOCSIS testbed behaves in a similar manner to the simulation model. The figure does suggest that a higher percentage of packets take advantage of piggybacking. We conjecture that implementation choices at the CMTS, possibly involving the MAP layout or some sort of optimization, explains the difference.

5.2.1.2 The TCP Throughput Workload

In the second study conducted on the testbed we analyzed the throughput of a single “always on” TCP application. In previous work we concluded that upstream channel of a DOCSIS system, because of its inability to return ACK packets at a high rate [29], can become a bottleneck in the downstream transport of bulk data. We created a single TCP flow in the downstream direction on the testbed and observed a maximum TCP throughput of 3.6 Mbps. We then simulated the identical network and also obtained a maximum downstream throughput of 3.6 Mbps.

5.2.2 Network 2: Charter Communications

We repeated the UDP echo experiments conducted in the DOCSIS testbed over a WAN with the Linux client located in a campus lab and the Linux server located on a residential network operated by Charter Communications. Charter’s service provided a 5 Mbps downstream rate and a 512 Kbps upstream rate. The interconnect between Charter and Clemson involved multiple backbone ISPs. As before the client sent a periodic stream of small (64 byte) UDP packets to the server which echoed each packet. We obtained a tcpdump trace at both the client and the server.

In the first test, the client sent a request every 0.002 seconds. Figures 11a through 11d show the distribution of the inter-packet departure (or arrival) times of both one-way streams at the respective sender and receiver sides. Figure 11a shows that approximately 97% of the inter-packet departure times are within 200 microseconds (the bin size) of their expected value. The jitter is attributed to random delays that occur in the operating system. Figure 11b shows that packets traveling over the path from the Clemson University network to the home network were subject to delay adding significant jitter to the stream. It is not possible to determine where the packet delay actually occurs.

Figure 11c shows minor additional jitter caused by processing overhead at the Linux server. Figure 11d shows the impact of DOCSIS on the upstream UDP flow. The upstream bandwidth consumed was 275 Kbps (accounting for headers) which does not overload the upstream channel. We confirmed this by verifying that minimal loss occurs. Based on our understanding of DOCSIS, Figure 11d suggests that Charter has configured its network for a MAP time of 2 milliseconds. The mode of 0.004 seconds
represents the two MAP times that are required to send upstream data. This is true even if piggybacking is used. Simulation experiments tell us that if concatenation is disabled, piggybacking would be used instead creating an arrival distribution at the sink consisting of a single mode at 0.004 seconds. The large mode at 0 seconds indicates that about 50% of the echo packets are being sent back-to-back over the upstream channel in a concatenated frame. Concatenated packets arrive at the client separated by the transmission time of the bottleneck link over the path between the CMTS and the client (which we estimate to be 45 Mbps).

Figure 11. Charter cable network observed departure and arrival distributions (2 ms packet spacing)
Figure 12. Simulation results (2 ms packet spacing)

Figure 13. Charter cable network observed departure and arrival distributions (5 ms packet spacing)
Figures 12a and 12b illustrate comparable simulation results. As in the simulation of the testbed experiments, we simulated only the echo reply stream by configuring the CM-1 node (shown in Figure 5) with a CBR traffic source that sent a 64 byte packet every 0.002 seconds. We enabled piggybacking and concatenation. To model the random delay observed in Charter’s network associated with UDP echo packets that arrive at the CM for upstream transmission (i.e., Figure 11c), we added an artificial delay to each packet before transmitting at the server. Based on the Anderson-Darling goodness-of-fit test the distribution illustrated in Figure 11c is neither normal nor Weibull (with 44000 samples and a 95% level of confidence the A-D statistic was an order of magnitude greater than the critical value). As a coarse approximation of the distribution we used a delay that is based on a normal distribution with a mean of 0 and a standard deviation of 0.0001. Figure 12a plots the inter-packet departure time distribution from the CBR source and Figure 12b plots the inter-packet arrival distribution at the UDP sink (S-1). Figure 12b shows the same bimodal result observed in Figure 11d. From this, we deduce that the Charter network uses a MAP time of 0.002 seconds and that it does support concatenation.

We next subjected the server in the Charter network to a stream of UDP packets in which a packet was sent every 5 milliseconds. Figures 13a through 13d show the packet arrival and departure time distributions for the echo request and reply streams. Figure 13d again suggests the network is configured with a MAP time setting of 0.002 because some number of UDP packets are sent 0.002 milliseconds after the main mode of 0.004 seconds. On an unloaded system, it takes one MAP time for the piggybacked request to arrive at the CMTS and then the subsequent grant to arrive at the CM. Figures 14a and 14b show the results of a comparable simulation run. The results are similar to the simulation associated with Figure 12 modulo the change in CBR traffic generator parameters. The CBR source is configured to add an artificial jitter based on a normal distribution with a mean of 0 and a standard deviation of 0.0005. To better fit the distribution illustrated in Figure 13d, we added 200 competing CMs that generated realistic amounts of Web traffic (primarily in the downstream direction) to the simulation. Comparing Figure 14b with 13d shows that the simulation model behaves in a reasonable manner. Without the competing CM traffic (but still with the CBR artificial jitter) the spread around the modes in Figure 14b was less than 200 microseconds. We conjecture that the differences are caused by the packing algorithms implemented by the Charter equipment and by the simulation. Figure 13d implies that the dynamics over the Charter network do not induce concatenation in this particular experiment while the simulation shows over 10% of all packets arrive in concatenated frames. More experimentation is required before we can draw conclusions from these results.

6. Conclusions and Future Work

We have presented the design and elements of the validation of an ‘ns’ DOCSIS simulation model. An analytic model was used to verify that our simulation operates correctly with respect to our understanding of the essential elements of the DOCSIS protocol. Further evidence that the simulation model is a valid representation of these elements of the protocol was obtained by running selected benchmark workloads on both the simulator and on operational DOCSIS networks.
DOCSIS is a large and complex protocol, and it is clear that aspects of the simulation are yet to be validated. In the next phase of this work we will use the testbed to show that the simulation model’s collision and recovery mechanisms work correctly in the presence of competing upstream traffic sources. When that effort is complete, UGS and rtPS service flows will be added to test QoS support. The final step in the evaluation process will be to use “real world” stochastically varying workloads.

Shared medium broadband access such as DOCSIS cable and WiMAX are complex. The behavior of these systems and their subsequent impact on the Internet is influenced by the network provider’s choice of equipment and by its configuration. The methods that we have used for model validation, particularly the analysis of live networks, help ground our simulation model to real systems. By making a validated simulation model available to the research community, we hope that others in academia also contribute to the evolution of DOCSIS broadband access networks.

### 7. Appendix A. Analytic Model

The maximum upstream application throughput, $T_{max}$, is defined as $D_{perCycle} / (t_{data} + t_{request})$. We derive $t_{request}$ for the case of no piggybacking, no concatenation, with a fixed number of contention slots allocated in each MAP, and with a backlog of packets waiting for upstream transmission.

![Figure 15. Backlogged contention request delays](image)

The $t_{request}$ has two components, the $D_{total RequestDelay}$ and the $D_{grant}$. The former is the total number of slots delayed from when a packet arrives until when the request for bandwidth is transmitted. For the backlogged case, this delay is simply the time from when the previous data transmission ends until when the bandwidth request is sent. The $D_{total RequestDelay}$ includes a static portion (the configured number of management slots $N_{ms}$) and a random portion (the collision backoff $D_{CSBackoff}$). The $D_{grant}$ represents the delay from when the CM transmits the request until it begins transmitting the data frame.

Figure 15 illustrates the $(j-1)^{th}$ MAP time during which the CM transmits data and also the next contention request. Once a packet arrives at the CM for upstream transmission, the CM randomly computes the number of contention slots it will observe before transmitting a bandwidth request. We represent this delay with the random variable $D_{CSBackoff}$ whose range is determined by the parameter $backoffRange^3$. The expected value is $E[D_{CSBackoff}] = (backoffRange) / 2$.

Depending on the value of the $backoffRange$ and the $N_{cs}$, the backoff delay might extend over multiple MAP times. The number of unused slots in a map is represented as $N_s = max(N_s - N_{data} - N_{cs} - N_{ms}, 0)$. In the backlogged case, the $D_{total RequestDelay}$ must consider three subcases depending on if the contention request gets sent during the current MAP, during the next MAP, or during a future MAP. We formulate the probabilities of each subcase.

**Case a.** If the selected contention slot is in the current MAP $mapTime_{j-1}$

\[ p_a = \begin{cases} 1 & \text{if } backoffRange \leq N_{cs} \\ \frac{N_{cs}}{backoffRange} & \text{if } backoffRange > N_{cs} \end{cases} \]

**Case b.** If the contention slot is in the next MAP ($mapTime_j$)

\[ p_b = \begin{cases} 0 & \text{if } backoffRange \leq N_{cs} \\ \min\left(\frac{backoffRange - N_{cs}}{backoffRange}, \frac{N_{cs}}{backoffRange}\right) & \text{if } backoffRange > N_{cs} \end{cases} \]

**Case c.** If the selected contention slot falls in a MAP at least 2 MAP times in the future ($mapTime_{j+1}$ or later)

\[ p_c = \max\left(\frac{backoffRange - 2N_{cs}}{backoffRange}, 0\right) \]

The expected delay from when the data transmission terminates until when the request is sent is the weighted delays for each subcase.

\[ E[D_{total RequestDelay}] = p_a[N_{ms} + E[D_{CSBackoff}]] + \\
\]
\[ p_b\left[N_{ms} + N_{cs} + N_s + N_{ms} + N_{cs} - \text{rem}\left(\frac{E[D_{CSBackoff}]}{N_{cs}}\right)\right] + \\
\]
\[ p_c\left[N_{ms} + N_{cs} + N_s + \text{floor}\left(\frac{E[D_{CSBackoff}]}{N_{cs}}\right)\times N_s + N_{ms} + N_{cs} - \text{rem}\left(\frac{E[D_{CSBackoff}]}{N_{cs}}\right)\right] \]

---

3 The actual range is $2backoffRange$ but for simplicity we use $backoffRange$. This parameter specifies the maximum number of backoff contention slots the CM must count before attempting a contention request.
The expected value of $D_{grant}$ consists of four delay components (all defined in units of slot times):

$$E[D_{grant}] = E[D_{RequestTrans}] + E[D_{CMTSMapAlignment}] + D_{schedule} + D_{MapTrans} + D_{CMMAP Align}$$

We summarize each component as follows:

- $D_{RequestTrans}$: Represents the transmission and propagation time experienced by the contention request. A bandwidth request frame fits in one slot which leads to

  $$D_{RequestTrans} = \text{cell}(t_{\text{slot}} + t_{\text{prop}}) / t_{\text{slotTime}}$$

- $D_{CMTS Map Align}$: Represents the number of slots from when the CMTS received a request from a CM until the beginning of the next MAP time. The value depends on if the contention request arrives during a MAP in which a data packet from the CM arrived (as in Figure 4) or if the request arrived during a MAP time which did not include a data grant. For the former case, the expected value is

  $$D_{CMTS Map Align} = N_{cs} / 2 + N_u$$

  For the latter case, the value is

  $$D_{CMTS Map Align} = N_{cs} / 2 + N_u + N_s - (N_{cs} / 2 + N_{mgt})$$

  The first case occurs with probability

  $$p = \min \left( \frac{N_{cs}}{\text{backoffRange}} - 1 \right)$$

  Therefore,

  $$E[D_{CMTS Map Align}] = p(N_{cs} / 2 + N_u) + (1 - p)(N_{cs} / 2 + N_u + N_s - (N_{cs} / 2 + N_{mgt}))$$

- $D_{schedule}$: Represents waiting time experienced by the request at the CMTS caused by scheduling delays. This delay will be a number of slots times that are integral multiples of the number of slots in a MAP. In other words, the granularity of the scheduling delays at the CMTS are in units of MAP times. For the analysis presented in this paper we assume that this delay is 0.

- $D_{MapTrans}$: Represents the transmission and propagation time (in slots) experienced by the MAP message in the downstream direction.

- $D_{CMMAP Align}$: Represents the number of slots from when the MAP message arrives at the CM until the grant. This will be

  $$D_{CMMAP Align} = N_s - D_{MapTrans}$$

8. REFERENCES


Dr. Jim Martin is an Assistant Professor in the School of Computing at Clemson University. His research interests include broadband access, autonomic computing, Internet protocols and network performance analysis. He has received funding from NASA, the Department of Justice, BMW, IBM, and Cisco. Dr Martin received his Ph.D. from North Carolina State University. Prior to joining Clemson, Dr Martin was a consultant for Gartner and before that a software engineer for IBM.

Dr. James Westall is a Professor of Computer Science in the School of Computing at Clemson University. His present research interests include computer network performance analysis and high performance computing using GPUs. He received his Ph.D. from the University of North Carolina at Chapel Hill.