User-Driven Dynamic Traffic Prioritization for Home Networks

Jake Martin
School of Computer Science
Georgia Institute of Technology
jake.martin@gatech.edu

Nick Feamster
Department of Computer Science
University of Maryland
feamster@cs.umd.edu

ABSTRACT

Network contention in a multi-user home setting can degrade performance for all participants. To maximize user experience, we propose that traffic be prioritized *based on the specific activity of the user*: the more a user interacts with an application and its associated traffic flows, the higher the priority that the application's traffic should receive. We introduce a client-side application that resides on the user's machine that monitors user activity and interaction with the application. This application sends information to the home router about user activity associated with different traffic flows; the home router then prioritizes traffic for flows that correspond to interactive traffic over those that are not. Finally, we introduce a protocol that allows the router and application to communicate. We show that the collective system improves the user experience for interactive applications by providing better performance for traffic associated with applications that a user is interacting with directly.

Categories and Subject Descriptors

C.2.6 [**Computer Communication Networks**]: Internetworking—*Routers*; H.1.2 [**Models and Principles**]: User/-Machine Systems—*Human factors*

General Terms

Algorithms, Human Factors, Management, Performance

Keywords

quality of service, traffic shaping, prioritization, home networks

1. INTRODUCTION

Home network applications range from simple Web browsing to streaming to backup. The variety of applications and diverse set of requirements for these network applications present challenges as these applications compete for limited network resources. To help combat the effects of home network congestion, many routers offer a means of traffic prioritization. Prioritizing network traffic from certain applications allows users to experience experience better network performance for those applications. Unfortunately, current methods of conducting this prioritization are static and coarse: they require the user to specify certain ports, applications, and traffic types that should receive higher priority. These priorities are static and are often based on the network port on which traffic is sent or received, rather than on whether the traffic flow corresponds to an application that the user is actively interacting with. We believe that traffic prioritization should not be static; rather, the network should prioritize traffic that is associated with applications that a user is actively interacting with.

The following example illustrates the need for dynamic traffic shaping. Consider a household where one user is uploading a large batch of photos to a popular social networking site, another user is downloading music, chatting with friends, and playing a video game, and a third user is are watching a video stream while playing a board game at the same time. While one user is uploading pictures, her operating system has decided it is time to download a large update; the second user is ignoring the music downloads and alternates between playing the video game and complaining to her friends in the chat program about the lag. Meanwhile, those playing the board game haven't been paying attention to the stream and are wrapped up in their game.

As humans, it is easy for us to see that the operating system update, the music download, and the video stream should be throttled: one user isn't concerned with updating her operating system, another more interested in gaming and chatting, and the computer showing the video stream might have even triggered a screen saver as a result of neglect. In short, some applications, such as the video stream and chatting, are interactive, in the sense that users may be interacting with the application as it is sending and receiving traffic. Others, such as the photo upload and music downloads are not. In this scenario, in the absence of other configured priorities, the interactive applications should receive a higher priority than the others to provide users with a better experience for users. Unfortunately, today's networks have no mechanism for dynamically providing higher priority to traffic that corresponds to interactive applications.

A router with static prioritization cannot account for these dynamic situations. On the other hand, if user activity is communicated to the router, the network can prioritize traffic according to the applications that matter the most to a user at any point in time. Even a sophisticated router with many conditional priorities based on user traffic will falter if different users prioritize the same situation in different ways. For example, if a static router presented with streaming traffic, a large download, gaming traffic, and general web browsing might have very comprehensive and convoluted rules regarding prioritized based on when and how the traffic is generated, where it is going, or from where it is coming, but the router cannot know which traffic the current users value and therefore which flows should be prioritized to maximize the users' experience.

In this paper, we develop a network control framework that dynamically prioritizes the traffic flows that are associated with the applications that a user is interacting with. The basis for intelligent dynamic traffic prioritization is to send signals to the router concerning which traffic flows correspond to user activity and which are background traffic flows. To ascertain user activity, we instrument the user's window manager to determine which window is in focus; we then assume that the window that is in focus is most likely the application the user is interacting with the most. Similarly, an active screen saver indicates a machine is not currently in use.

After determining which traffic flows correspond to the application window that is in focus, the operating system sends messages to the router that instruct it to prioritize the traffic flows that correspond to the active application. The router receives this information, parses it, and assigns priorities based on whether an application is active (in focus) or inactive (in the background), as well as whether the host itself is active. Traffic flows associated with an active application on an active host receive higher priority (and therefore better performance) than applications running in the background or on an unattended machine.

Beyond the immediate practical use of improving network performance, the system can be used for gathering information about network activity and uses in the home. The boost in network performance provides an incentive for users to allow their network usage habits to be logged. Logging the combined activity data and network metrics for home environments presents unique opportunities for other longitudinal studies, as well as developing better methods of detecting user activity.

The rest of the paper is organized as follows. We discuss related work and similar endeavors in Section 2. Section 3 describes the system design and implementation in more detail. Section 4 describes possible enhancements to the system, and Section 5 concludes.

2. RELATED WORK

Industry and academia have both seen much research into different methods of traffic shaping and providing quality of service (QoS) to end users on home networks. Unfortunately, none of the current options for providing traffic prioritization on home routers provide the flexibility that allows traffic prioritization to adapt to users' changing needs. In other cases, the solutions rely on applications to determine their own priority; such an approach is inadequate if the user is not actually paying attention to the application in question. Solutions that have overcome either of the first two hurdles often are prohibitively difficult to implement in an actual home network environment.

Traffic prioritization at the end host Previous research examines providing dynamic quality of service to *applications*, as can be seen in Brandt *et al.*'s work [1,2]. Although this system allows applications to dynamically receive different priority levels, the implementation is not practical in a home network. The system focuses on achieving complete utilization of the host's network resources, and all prioritization decisions are made on the end host, rather than on the router. Brandt's methods could be extended by shifting resource allocation to the router, but the existing design would still require extensive communication between hosts and the router, which might prove to be cumbersome in such a heterogeneous environment as a home network.

Traffic prioritization at the router Liu *et al.* propose to achieve traffic prioritization in a home network using an intelligent router that analyzes upstream and downstream traffic and makes prioritization decisions based on flow properties [10]. The approach manages traffic for a home network, but decisions are based only on traffic classification by overriding the type of service (TOS) byte in the IP header. This approach also places the burden of determining the priority of traffic flows on the applications themselves, which makes it difficult to manage *all* flows coming from all devices is in home network. Furthermore, the approach provides scant security or resource protection, since nothing prevents applications from setting the TOS byte to the highest priority, effectively giving all such applications equal priority and circumventing QoS.

Zinca *et al.* studied how to augment the network with host-based quality of service agents that can communicate with a central QoS manager to deliver a certain level of priority to sets of applications (e.g., multimedia or VoIP applications) [5]. This architecture inspires our approach; however, the proposed system still allowed individual applications to determine their own priority levels, independent of whether the application actually required a certain level of service. Therefore, the proposed system is still vulnerable to applications that might greedily request a higher level of prioritization even when the users themselves do not benefit from receiving a higher level of priority for that application. In contrast, we develop a low-level monitoring agent that applications cannot directly influence; this agent reports only system state information, leaving a central controller to determine the priority that each flow should receive based on the corresponding state of the application. In this way, we prevent rogue or greedy applications from diverting resources from applications with which a user may be interacting.

Holistic approaches to traffic prioritization Katchabaw *et al.* developed a quality of service system that uses hints from individual hosts to make decisions about traffic prioritization [7, 8]. Our work draws inspiration from and builds on Katchabaw's system, but our solution has been designed specifically to overcome potential barriers to deployment.

First, Katachabaw's approach requires many aspects of the network—including the end-host, the network routers along the path, and the remote servers—to be involved managing traffic priority. The heterogeneous nature of today's

Internet and other networks (at large and even within a household) means that implementation of a solution as involved as that proposed by Katchabaw would be slow at best. In contrast, our solution can be deployed in a home network without modifications to any infrastructure outside the home. Hence, our approach can be deployed independently in homes today.

Second, the system relies on priority policies set by applications, users, or operators in addition to signals from the end host's window manager. This approach requires users to be aware of the system in order to realize the benefits that the system provides. In most home environments such a system is infeasible, both because there are few current applications which are capable of adhering to his system and because most home users are unwilling (or do not have the knowledge) to specify such policies. In contrast, our system develops a more transparent solution that does not need to be understood by any user-facing application or network user. Although our system does offer only coarse-granularity prioritization, it is both application agnostic and easily configured by a typical home user.

A final important tenet of Katchabaw's work is that *to be effective, QoS must be reflected in the network, CPU, and all resources for which processes compete.* Although we do not directly challenge this assertion, we feel that a network-only solution would still benefit home network users. To extend the system to control CPU usage or other low-level resources would mean tighter integration with hosts, which increases complexity. Instead, our solution is no more complex and is as deployable as the current configuration software packaged with many home routers today.

Studies of congestion and contention in home networks Other projects, such as HomeMaestro [6] and Home Watcher [3], have studied the home network performance and the effects of congestion on these home networks. Home Watcher, addresses the problem of home network congestion and its effects on users. HomeMaestro was developed to identify network problems and conflicts. Solving the networking issues through prioritization falls outside the scope of HomeMaestro's goals. HomeMaestro does describe how a traffic prioritization scheme could easily use the information generated by their software, and our approach could incorporate cues from HomeMaestro to enhance our heuristics. Similarly, Home Net Profiler provides valuable information, both for researchers and end users, regarding the state of the home network [4].

3. DESIGN AND IMPLEMENTATION

The system system has two components: (1) the router that receives and processes traffic prioritization hints; and (2) the client software that runs on each host that associates applications with user activity and subsequently assigns associates certain traffic flows with higher priority levels. The client software infers and reports the application with which the current user is interacting. The router processes this information and assigns priorities and network resources accordingly.

3.1 Router

The router's main task is to give preference to traffic flows for which users will notice poor performance, at the expense of flows that users will not notice a degradation in performance. We assume that a user would benefit the most by having the application with which they are currently interacting receive the highest priority. We also assume that a user will be indifferent to the level of priority that traffic flows from a background process or inactive end host ultimately receives.

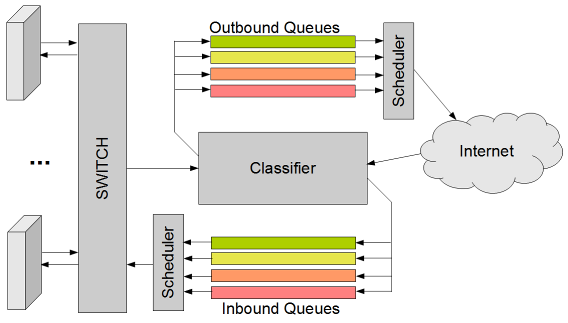

Figure 1: Design and implementation of the router that achieves dynamic traffic prioritization given hints from end hosts about priority.

Byte Offset	0	1
0	UDP HEADER	
4	Activity	num_ports
6	port_1	
8	port_2	
	...	

Figure 2: The control packet is delivered over UDP. The first four bytes are the UDP header, the next two bytes indicate activity and the number of ports n. The next n bytes are the ports associated with the application's traffic flow.

Figure 1 shows the design of the router, which we implemented in Click [9]. We used a standard `EtherSwitch` to provide basic switching. We then added classification and priority management logic, as well as a custom lottery traffic scheduler. The router achieves user-driven dynamic traffic prioritization using three main steps: classification of packets, priority management, and delivery of packets. We describe these steps in more detail below.

Step 1: Classify Packets Control packets originate from the clients installed on the hosts and are sent to port 6250 – all other packets on other ports are normal traffic. Control packets are sent over UDP as there is little harm from packet loss; the current application simply does not receive top priority. After a control packet is identified, the UDP payload is examined for application usage data. The first byte indicates the activity of the host: 0 for inactive, 1 for active. The second and third bytes inform the router how many ports are associated with the active application (e.g. browsers implementing HTTP pipelining). The remaining bytes in the control packet are a listing of all of the ports used by that application using 2 bytes per port (see Figure **??**).

The port numbers provided are the source port number used by the sockets opened by the application; these are

		Application	
		Foreground	Background
Host	Active	40	25
	Inactive	20	15

Table 1: Of 100 tickets, these table illustrates how many tickets each type of traffic flow will receive depending on the users' interaction with the application.

unique per host and combined with the host's IP address serve as a unique identifier for each application's associated traffic flows.

Step 2: Manage Priorities After the router classifies a traffic flow, it must assign a priority to that flow. If the packet is a control packet, the router extracts the IP address and active ports and updates the priority management data structures accordingly. To maintain a complete list of active and inactive hosts and inactive and active applications, the router maintains two tables: one for active hosts, and one for inactive hosts using the IP address as the key. The value at a particular address is a bit-string corresponding to the activity levels of each port: 0 for inactive, background ports (or unused ports), and 1 for active, in-use ports.

Adding a new host with a new active application is simple: the router creates the port bit-string by setting active ports to 1, then inserts the bit-string into the active application's hash table. To modify an old entry, the router simply overwrites the old bit-string with the new one. The previously active application has been closed or sent to the background, so it is now an inactive application on an active host. Because all inactive applications are treated in the same fashion, the router only needs to track the one current active application. Inactive hosts can be updated in the same way. After the priorities have been updated, the control packet is dropped.

If the packet was not a control packet (i.e., it was regular traffic), then the first step is to identify if the packet came from an active or inactive host. Hashing the IP address and looking up the appropriate bit-string in the active and inactive bitmaps will identify if the host is active or inactive. In the next step we compare the source port found in the appropriate transport layer header to the ports found in the bit-string; then if no source port is found or the source port is not found in the bit-string of active ports, the packet is classified as belonging to an inactive application. The packets are then placed into their respective queues: activeAppActiveHost, inactiveAppActiveHost, activeAppInactiveHost, inactiveAppInactiveHost.

Step 3: Forward Traffic The router uses a combination of lottery [13] and round-robin scheduling to select which queue to service at any point in time. An active application queue is given more tickets than an inactive application queue, and the same holds true for active host queues over inactive host queues (see Table 2). The exact ratio of tickets can be changed, but currently an active application has between 1.3–1.6 as many tickets as an inactive application and an active host has 1.6–2 as many tickets as an inactive host.

Tickets are drawn randomly, however should the winning queue be empty, the scheduler checks adjacent queues until a packet is found or all queues have been checked. Using a lottery scheduler prevents starvation, which precludes an active application on an active host from receiving all available bandwidth. The addition of the round-robin elements prevents wasted lottery cycles. For example, if a ticket designated for the inactiveAppInactiveHost queue were drawn and there were no inactive hosts, then without the round-robin elements, the round would be wasted and a packet would not be sent. The modified lottery scheduler combined with round-robin elements provides full utilization of the networking resources without starving lower priority applications and affecting overall user perception of network performance.

3.2 Client

The client application residing on the end host currently instruments the window manager to find the active application using `xdotool`. For the purposes of this paper, we assume that the active application is the one that has focus in the window manager. Because the client uses window focus to determine priority, the current prototype client agent is not compatible with command line processes or with unsupported window managers. After finding the active application, the agent uses `lsof` to match the process with its associated sockets. It then crafts a control packet as defined in Figure 2 and sends the control packet to the router over the corresponding socket.

Currently, the client periodically runs a function to determine the application that has window focus every few seconds. Refinements of this agent might ultimately use window-manager events, and send updates only when the active window changes or when a host has been inactive for a certain amount of time.

4. FUTURE WORK

Before our proposed system can be viable, the routing logic and prioritization mechanics must be moved from a user-space Click implementation to a home router. The BISmark platform [12] is a programmable router platform that supports OpenFlow control [11], and may ultimately serve as a possible deployment platform. We expect that a router-level implementation will avoid much of the overhead that is incurred with the current implementation in a user-space Click program.

We must also perform additional testing to determine the most effective queuing discipline for achieving dynamic prioritization. We chose the lottery scheduler largely because it was easy to implement and integrate with Click. A more thorough examination and evaluation of different queueing disciplines is warranted.

Our method for identifying the software with which the user is interacting is somewhat primitive, and developing better methods to determine users' level of interaction with each application (and, hence, the priority level the application's traffic should receive) is a ripe area for future research. A first step for improving the current approach would be a better integration with the window manager, to provide event-driven updates to traffic priorities, rather than having the client periodically poll the window manager. Additionally, there may be other ways to more accurately determine applications that a user is interacting in addition to using window focus alone. Explicit polling of user activity or better inference techniques for determining which applications may still require high priority even though the application window is not in focus may ultimately improve the user experience. Developing an activity profile for a user that determines the applications that any given user tends to use—and whether they care about interactivity for each of those applications—can also ultimately improve the user experience.

One possible method for ascertaining user activity could combine the data gathered from the window manager with data from a camera (e.g., the user's Web cam). Using established methods for tracking eye movement, the client might determine where on the user's screen that user is looking. When combined with the layout of the user's open windows, the client could infer the applications that a user is paying attention to, even if the user isn't technically interacting with the application with a keyboard or mouse (e.g., if a user is watching a stream on half of the screen while writing a paper on the other half).

We would be remiss to ignore the potential security concerns of transmitting the applications a user would be using and when. Ultimately, the fact that this prioritization occurs exclusively within the home network offers some level of privacy, but a future implementation of our system should encrypt the control messages that a client sends to the router regarding the prioritization of application traffic. Ideally, these control messages would also be signed by the clients running on end hosts to ensure that the control messages are not modified in flight by an adversary who wishes to affect how the home router prioritizes traffic.

5. CONCLUSION

Despite ample research on traffic shaping and quality of service, relatively little of that research has been focused on user experience with interactive applications, particularly in a home network setting where priorities may evolve rapidly. The system we have described in this paper relies on hints from end hosts to determine the applications that a user is likely to demand high priority for at any point in time; the home router uses these hints to assign a higher priority to certain applications. Indicating to the router the applications that a user is using to the router allows the router to prioritize traffic flows for applications that a user is interacting with at any point in time.

To achieve a fair, dynamic allocation that responds to user activity without starving inactive hosts, we implemented a lottery-based scheduler with round-robin elements in a Click router. The key to increasing user experience with our system is a complete, accurate picture of how the user is interacting with each application that is sending traffic on the home network. Our current implementation provides that information to the router by instrumenting the end host's window manager and instructing the router to prioritize traffic flows for the application whose window is in focus. Other means of inferring user interactions, such as actively tracking eye movements, may further improve the user experience.

Acknowledgments

This project was supported in part by NSF CAREER Award CNS-0643974, NSF Awards CNS-1040715 and CNS-1059350 and a Google Focused Research grant. Nate Barish, Curtis Free, Nigel Lawrence, and Paul Plunkett helped implement the system and write an earlier draft of this paper.

REFERENCES

[1] S. Brandt, G. Nutt, T. Berk, and M. Humphrey. Soft real-time application execution with dynamic quality of service assurance. In *Quality of Service, 1998. (IWQoS 98) 1998 Sixth International Workshop on*, pages 154 –163, may 1998.

[2] S. Brandt, G. Nutt, T. Berk, and J. Mankovich. A dynamic quality of service middleware agent for mediating application resource usage. In *Real-Time Systems Symposium, 1998. Proceedings., The 19th IEEE*, pages 307 –317, dec 1998.

[3] M. Chetty, R. Banks, R. Harper, T. Regan, A. Sellen, C. Gkantsidis, T. Karagiannis, and P. Key. Who's hogging the bandwidth?: The consequences of revealing the invisible in the home. In *CHI 2010, Association for Computing Machinery*, April 2010.

[4] L. D. Cioccio, R. Teixeira, and C. Rosenberg. Measuring and characterizing home networks. In *Proceedings of ACM SIGMETRICS*, June 2012.

[5] D.Zinca, V.Dobrota, C.M.Vancea, and G.Lazar. Protocols for communication between qos agents: Cops and sdp. In *COST #276 Workshop on Information and Knowledge Management for Integrated Media Communication*, 2002.

[6] T. Karagiannis, C. Gkantsidis, P. Key, E. Athanasopoulos, and E. Raftopoulos. Homemaestro: Distributed monitoring and diagnosis of performance anomalies in home networks. In *MSR-TR-2008-161*, 2008.

[7] M. J. Katchabaw, H. L. Lutfiyya, and M. A. Bauer. Using user hints to guide resource management for quality of service. `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.34.3136`, 1999.

[8] M. J. Katchabaw, H. L. Lutfiyya, and M. A. Bauer. Usage based service differentiation for end-to-end quality of service management. `http://www.csd.uwo.ca/~katchab/pubs/comcomjournal.ps`, 2005.

[9] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The Click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, Aug. 2000.

[10] G. Liu, S. Zhou, X. Zhou, and X. Huang. Qos management in home network. In *Computational Intelligence for Modelling, Control and Automation, 2006 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on*, page 203, December 1 2006.

[11] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. OpenFlow: Enabling innovation in campus networks. Apr. 2008.

[12] S. Sundaresan, W. de Donato, N. Feamster, R. Teixeira, S. Crawford, and A. Pescapè. Broadband internet performance: A view from the gateway. Toronto, Ontario, Canada, Aug. 2011.

[13] C. A. Waldspurger and W. E. Weihl. Lottery scheduling: Flexible proportional-share resource management. In *1st USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pages 1–11, 1994.