PERM: A Collaborative System for Residential Internet Access

Technical Report UIUCDCS-R-2006-2751

Nathanael Thompson and Haiyun Luo
Department of Computer Science
University of Illinois at Urbana-Champaign
{nathomps,haiyun}cs.uiuc.edu

Abstract

802.11 networks have spread rapidly in residential areas, and it is common for neighbors to receive the signals from each other's home wireless routers. Residents can leverage such an opportunity to improve their Internet connectivity, at no additional cost, by pooling their individual Internet connections together. In this paper we present our design, implementation, and evaluation of PERM, a system that enables collaborative Internet access in residential areas. Compared with related work, PERM is practical in that it does not rely on support from the network infrastructure in terms of advanced naming or proxy, or from the remote host in terms of new transport protocols. Instead, PERM employs automated on-line analysis of the user's networking behaviors, and exploits the recognized patterns to achieve high-performance scheduling at the flow level. PERM is also highly usable for normal residential users. It mitigates the free-riding problem, and preserves a user's privacy and security. We have implemented PERM in the open-source LinkSys wireless router and Linux client. Our evaluation in a dual-homed residential host shows that PERM significantly improves perceived last-mile Internet connectivity at homes.

1. INTRODUCTION

Internet applications are always pushing the limits of the last-mile residential Internet connectivity. Despite the improvement offered by DSL and Cable over traditional dial-ups in the US, the last mile Internet access at homes is hardly adequate to support multiple VoIP, P2P file sharing, swarming downloads, and/or web transfers. In addition, existing broadband access often suffers from temporary outages. Those outages usually leave individual users or an entire community disconnected for hours or even days[1], which is less and less tolerable as people's daily lives depend more and more on the Internet.

In our waiting for the next major update on the capacity and resiliency of the last-mile Internet connectivity, several facts caught our attention. First, the broadband penetration ratio has been growing steadily over the last several years. By March 2006 over 69.4% of American Internet users subscribe to some form of broadband service, and over 75% in the UK [2]. The density of broadband users is already high. Second, broadband access in residential areas has diversified to the point that many residents have a choice from several broadband ISPs serving the same area with DSL, Cable, Satellite, or emerging WiMAX. There is no clear domination in the US local ISP market. Third, accompanying the spread of broadband is the deployment of the 802.11 home wireless networks. A January 2005 survey by Parks Associates found that 52% of US households with a home network were using 802.11. Given the communication range of 802.11 devices it is common for a resident to receive the signals from his neighbors' wireless routers. Finally, the broadband Internet access is always on but seriously under-utilized. Recent surveys show that residential users are online for an average of only 12.7-14.7 hours per week [1].

[1]Typical broadband ISPs only guarantee the recovery of Internet connections in two to three business days.

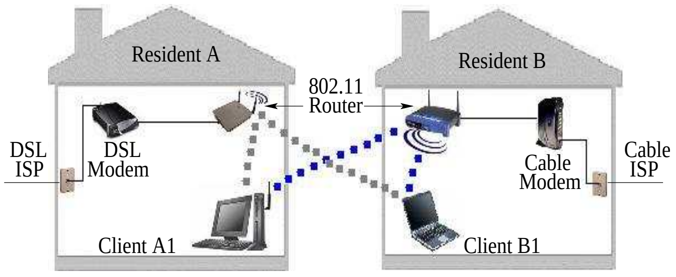

Figure 1: 802.11 home networks enable collaborative Internet access in residential areas.

These observations suggest that there is a great opportunity for neighbor residents to share their broadband Internet connections, through high-speed first-/last-hop wireless networks, for the mutual benefit of improved Internet connectivity. The improved Internet connectivity will translate to both higher bandwidth and higher resiliency, perceived at each participating resident. Figure 1 shows one example where two neighbors, subscribing to DSL and Cable respectively, schedule flows across each other's wireless routers. Since it is unlikely that two neighbors are always active at the same time, one can significantly increase the performance of his networked applications and availability of the Internet connectivity by spreading traffic between the two available Internet access networks (Cable and DSL), as a result of the increased statistical multiplexing gain. More importantly, no additional monetary cost beyond the existing broadband subscription and a second 802.11 wireless card is involved. The entry bar is very low for a residential user to participate in such a collaboration.

In this paper we present our design, implementation, and evaluation of PERM, a system that enables automated collaborative Internet access in residential areas. PERM addresses the following three major challenges to realize the above vision. First, PERM targets incremental deployment by normal individual residential users. Therefore the system must be contained within a user's home. This requirement rules out the majority of the related work, such as Internet multihoming [10, 18, 20, 32] and advanced routing [12, 19, 11, 26, 13] that rely on support from the network infrastructure, or advanced transport [22, 15, 21] and flow migration [23, 25, 29, 30, 24] that rely on the support from the other end of the connection. Since packet striping or flow migration needs resources that are generally not available to a resident, we chooses to schedule flows[2] across available Internet connections in PERM. That is, PERM only enforces its scheduling control at the flow establishment phase. To compensate the loss of the scheduling flexibility, PERM exploits its proximity to the end user. It adopts automated on-line learning of the end-user's traffic patterns to make the best match between flows and available Internet connections [33]. Again for the practical acceptance by individual residential users, the PERM implementation is transparent to both applications and the OS kernel. No update or recompilation of legacy networked applications or OS kernel is necessary.

Second, a selfish user may take advantage of PERM to free-ride over his neighbors' Internet connections without contributing his own. This free-riding problem, common in peer-to-peer resource sharing systems in general, may discourage compliant users from participating. To mitigate the free-riding problem and promote fair sharing, we build in PERM two mechanisms, namely certified identity control and adaptive symmetric crediting. Certified identity control enforces a non-trivial cost of obtaining a new identity, while adaptive symmetric crediting gradually increases the level to which two PERM users share their Internet connections, as their mutual trust builds up over time. The sharing level is defined as the amount of initial credits and the bandwidth cap for a recognized PERM neighbor.

Finally, PERM must preserve the security and privacy of a user's Internet traffic at the same level as without PERM. Since WPA (or the next update to 802.11i) defeats an eavesdropper readily, PERM only needs to protect the security and privacy of a user's networked behaviors against his collaborative PERM neighbors. Note that even if the traffic is end-to-end encrypted, e.g., through SSL/TSL, the IP address of the other end of a connection could be considered as part of a user's privacy. Although the desired privacy could be achieved by routing packets through a high capacity, trusted relay proxy through a secure tunnel, such proxy resources are unlikely to be available for every residential user. PERM instead selectively schedules flows to avoid security and privacy compromise. In specific, PERM allows the user to specify the set of Internet networks, e.g., the network addresses of known financial, medical, adult, and government agency web sites, with which the connections should always be scheduled through his own wireless router and ISP. Furthermore, a user specifies the set of networks, e.g., his own home network, to which the access requests from neighbors will be denied.

[2]In our implementation a flow is identified by a [SrcIP, SrcPort, DstIP, DesPort] tuple.

We have implemented PERM in the Linksys WRT54G wireless router and Linux clients. At the client side a PERM scheduler is responsible for scheduling flows based on performance optimization, incentive availability, and security and privacy policies. At the wireless router side we deploy a PERM incentive and flow manager that manipulates an onsite RADIUS service module to deal with free-riding, iptables filters to enforce privacy and security, and a class-based QoS scheduler to limited the traffic rates from neighbors, depending on their credit availabilities.

The PERM system is highly flexible and extensible. At the higher usage level explicit configuration and control on incentive, security, and privacy are enabled at both the client and the wireless router through a web interface, similar to the way existing home wireless routers are configured. However, we anticipate that the default settings will meet the needs of most residential users. At the lower level critical components, such as the scheduler and the monitor, can be easily upgraded or replaced to accommodate individual requirements or future evolution. To deploy PERM a user only needs to install the client side scheduler and change the firmware on his wireless router, two steps which involve no additional monetary commitment. PERM will maintain incentive, enforce security and privacy, enforce priorities, evaluate the instantaneous quality of accessible Internet connections, characterize application traffic, and schedule flows accordingly. To the best of our knowledge, PERM is the first practical system that includes all necessary functionality for practical collaboration in residential Internet access.

The rest of this paper is organized as follows. In the next section compare with the related work. We describe various aspects of PERM system design in Section 3, and describe the details of our prototype implementation in Section 4. The performance evaluation is presented in Section 5 followed by discussions on PERM's impact on ISPs in Section 6. Finally the paper concludes in Section 7 with future work.

2. RELATED WORK

In this section we compare with existing work on route selection, Internet multihoming, incentive management, bandwidth, and flow migration.

Route selection is probably the closest to PERM in that the main function of a PERM scheduler at the client side is to route flows across different neighbors' ISPs. Existing route selection is usually achieved by either source routing (SOSR [19], Nimrod [13], IP Loose Source Routing), proxy routing (MONET [12]), or overlay routing (Detour [26], RON [11]). However, the application of those mechanisms in PERM will require either advanced support from the routers (IP source routing), the deployment of routing proxies beyond the bottleneck Internet access links (proxy routing), or Internet intermediary infrastructure (overlay routing). It is unlikely that any of those resources will be freely available for each residential user in the near future.

Internet multihoming is also relevant to PERM in that a PERM client indeed multihome across neighbors' Internet connections. Akella *et al.* first show that more than 40% improvement can be achieved using multiple ISPs, assuming perfect knowledge about the providers and ability to change routes arbitrarily [8]. Following work [9, 32] reveals that multihoming could achieve the same level of performance as overlay routing. Experiments on enterprise multihoming [10] provide empirical performance studies of NAT-based in-bound ISP link selection. Guo *et al.* summarize mechanisms for enterprise multihoming load balancing in [20]. Those mechanisms, valid for multihoming at the AS or enterprise level, do not directly apply in end-host residential systems like PERM, where the availability of advanced router or proxy supports cannot be assumed.

PERM incentive management is related to MoB [14], a collaborative infrastructure for wide-area wireless data services. Different from MoB where an eBay-alike centralized trust management system is maintained, PERM is distributed and complies with its spirit of peer-to-peer Internet access sharing. PERM's incentive management model is similar to BitTorrent's tit-for-tat [16].

PERM enables opportunistic aggregation of the bandwidth of the neighbors' Internet connections, a topic that has been intensively studied in other contexts. For example, link layer bandwidth aggregation coordinates multiple underlying radios and present a virtual interface to higher levels [6, 7, 28]. They do not apply in PERM where the bottleneck Internet access links belong to different ISPs. Multiple access links can be leveraged through bandwidth aggregation at transport layer [21, 22, 30]. However, it requires modification at both ends of the connection. Another approach relies on the deployment of intermediate proxies (e.g., aggregation proxy in [27], and Mobile-IP agent in [15]). It again relies on advanced network middleboxes and their appropriate placement.

Flow migration enables fine time granularity scheduling. However, existing Internet flow migration proposals have the same issues as bandwidth aggregation techniques do. SCTP [30] is a transport layer solution in which both ends share information about available interfaces and agree to migrate connections. MSOCKS [23], MAR [25], and Migrate [29] are in some ways similar to Mobile IP, using middleboxes to achieve flow migration. The Host Identity Protocol incorporates a new naming scheme for the Internet, which would allow hosts to communicate over more than one interface [24].

3. PERM DESIGN

PERM enables residential collaborative Internet access through *incentive and flow management*, *security and privacy enforcement*, and *high-performance flow scheduling*. A scheduler on each participating client assigns flows to the available Internet connections. At each wireless router an incentive manager runs to ensure that neighbors are sharing Internet connections fairly. Besides the access control based on incentive management, the wireless router also does traffic shaping to differentiate the flows initiated by the owners and credited/uncredited neighbors. Finally privacy policies are enforced by the schedulers at both the client and the wireless router. In this section we start with the system model, and then describe our designs for these three functions one by one.

3.1 System Model

We consider Internet access in residential areas, as illustrated in Figure 1. A compliant residential user subscribes to a certain form of Internet access, e.g., DSL or Cable. The user shares his Internet access among his own client devices (home clients) and neighbors' devices (foreign clients), through a high-speed wireless router (home wireless router) as the gateway to his ISP network. A PERM incentive and flow manager is built in the user's home wireless router. PERM schedulers are installed in the home clients, e.g., home PCs. The incentive and flow manager works in tandem with these schedulers (home schedulers), and interacts with the schedulers at neighbors' client devices (foreign schedulers). Similarly, a scheduler coordinates with the flow manager at the home wireless router (or home flow manager), and interacts with neighbors' wireless routers (foreign wireless routers). We assume multiple 802.11 wireless interfaces are installed at a client, so that it simultaneously connects to both the home wireless router and one or more foreign wireless routers. Note that since the majority of laptops already have internal 802.11 interfaces built in, they just need a second PCMCIA or USB 802.11 interface to operate in PERM. A client can also be attached to the Ethernet port on the home wireless router.

3.2 Incentive management

PERM can be successful only if neighbors are willing to share their Internet access with one another. Most users will not if they believe they are not receiving fair trade for the Internet access they offer. To promote fair sharing PERM uses a credit based incentive model as the default incentive management policy, similar to that in BitTorrent [16]. From a residential user's perspective, his client devices consume credits when the schedulers route traffic through foreign wireless routers, while his home wireless router earns credits when it forwards packets for foreign clients. A certain number of initial credits, e.g., 100K bytes, is granted to each new neighbor to bootstrap the sharing.

Each wireless router employs class-based priority scheduling to control traffic at the wireless router. There are three types of clients that could connect to a wireless router: *home clients*, *credited foreign clients*, and *uncredited foreign clients*. Home clients are the devices belonging to the owner of the wireless router. Foreign clients are those belonging to neighbors. The traffic initiated by home clients is always treated with the highest priority in order to maintain good performance for the owner on his own wireless router. It is never bound by credits. In contrast, the traffic initiated by foreign clients is rate-limited. A credited foreign client, i.e., the foreign client with credits more than a minimum threshold, will have its traffic capped at certain rate unless the home clients are idle. A uncredited foreign client, i.e., the foreign client with credit less than the threshold, will have its traffic bound to a smaller rate, unless both the home and the credited foreign users are idle.

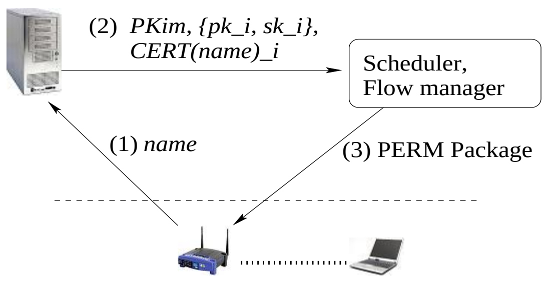

Figure 2: Certified identity control

One subtlety of above crediting strategy is that the maximum traffic volume a user A can send through a neighbor B's wireless router, before user A uses up its credits and is sentenced by user B as a uncredited foreign client, is bounded by the sum of A and B's initial credits at each other's wireless routers. To maintain high-performance Internet connection when a user's own ISP is in outage, the initial credits would have to be large, opening the door for free riders as described below. PERM allows a user to assign a relatively small initial credits to a new foreign client, and progressively increases the initial credits as their interactions build up. This mechanism is particularly useful for users whose ISPs charge not by a flat subscription fee but by actual usage. Those users tend to be more cautious. They route large amount of traffic of a neighbor only if they are highly confident, through the history of sharing, that the same amount of traffic can be routed through the neighbor' wireless router at higher speeds in the same billing period. They can also opt to set the rate limit for uncredited neighbor clients to zero, essentially denying the Internet access from uncredited neighbors. PERM allow the owner to change the default initial or the current credits of foreign clients.

As a general requirement for both incentive management and security and privacy enforcement (described in the next section), a user must identify the wireless router and all the clients that belong to the same neighbor. Furthermore, a selfish user can manage to free ride on neighbors' Internet connection over long term by frequently changing his identity after the initial credits are consumed, a behavior similar to "Sybil Attack" [17] that is generic in distributed p2p resource sharing systems. One approach to user management would be to have a centralized authority generate names and authentication credentials for each user. The authority could be global, or even local. However, one goal of PERM is immediate deployment within the existing infrastructure. The overhead and commitment required to support such an authority would delay its establishment and hinder the spread of PERM. We therefore resort to automated, distributed mechanisms.

The main mechanism we build into PERM is *certified identity control*, as illustrated in Figure 2. Similar to the trusted-third parties' (or Internet central authorities') public keys that are embedded in every web browser, the public key PK_{im} of a PERM identity manager is built into each PERM distribution. In specific, on request we first acquire a unique *name* from a user. We then generate a public-private key pair $\{pk_i, sk_i\}$ called "user keys", and sign the binding between the name and the public key pk_i in a certificate $CERT_i^{name}$ using identity manager's private key SK_{im}. The user will then obtain the PERM package, including the scheduler and the wireless router software, with the identity credentials PK_{im}, $\{pk_i, sk_i\}$, and $CERT_i^{name}$ embedded. These credentials will be applied in the mutual authentication between a client and a home/foreign wireless router through standard WPA (or 802.11i). More details of the authentication are presented in Section 4.1.

As the default PERM identity manager we hold the private key SK_{im} in secret. Three policies are employed to confine free riding. First, the identity, i.e., $\{pk_i, sk_i\}$ and $CERT_i^{name}$, is generated for each request, and will always be distributed in a complete PERM package. Although we allow a user to install PK_{im}'s of other identity managers so that he can choose to honor the identities generated by identity mangers other than his own (e.g., the one that manages his local community), by default we do not expose the interface for a user to import new identities into his existing PERM installation. Second, over a certain period, say one month, we only distribute one identity to a specific IP address. Finally, PERM delays sharing Internet access with a new neighbor after his first appearance, and starts the sharing with small amount of initial credits. These mechanisms do not completely eliminate free riding of a sophisticated hacker. However, it does add a significant cost (new download, software installation, and extra latency) to change the identity.

For a specific download request, the certificates that are embedded in the scheduler software and the wireless router image will have the same unique *name* supplied by the user. This way, we bind together the client devices and the wireless router that belong to the same user. With the same name the scheduler at a client and the flow manager at the wireless router will be easily identified when interacting with those of another neighbor. A credit account will be maintained for each certified neighbor. Depending on the traffic volume, credits will be added to the neighbor's account when his wireless router forwards packets for the user, and be deducted from the account when the user's wireless router serves any of the neighbor's clients.

3.3 Security and privacy policy

A second goal of PERM is to preserve the security and privacy of users' traffic at the same level as without PERM. With certificates established as described in the above section, PERM enables WPA to authenticate users and encrypt the wireless connections to defend against external eavesdroppers[3]. Therefore, the focus of PERM security and privacy enforcement is to defend against the neighbors with which the user shares Internet access.

With users identified through certified *name*s, a wireless router is able to differentiate the traffic of the home clients from the traffic of the foreign clients, and apply different controls accordingly. To protect the home network the wireless router will not allow direct communications between a home

[3]Note that authenticated users cannot sniff each other's traffic because WPA establishes per-user keys to encrypt the traffic.

client and a foreign client. Furthermore, connection requests from foreign clients to sensitive destinations, such as financial and medical institutes, adult websites, and governmental agencies, can be blocked based on their hierarchical DNS names and IP addresses. Note that above settings are the defaults, expected to be appropriate for the majority of residential users. Individual users can configure those rules at their home wireless routers based on their own preferences.

Similarly, a scheduler located at a client differentiates between home and foreign wireless routers identified by their certified names, and schedule the outgoing traffic from the client accordingly. The scheduler will always schedule the client's flows with sensitive destinations to the home wireless router, the same way a home wireless router blocks connections between sensitive destinations and foreign clients. Furthermore, connections that are end-to-end encrypted, e.g., SSH and TLS (https), are likely privacy sensitive. The scheduler at the client routes such connections, recognized by their port numbers, to the home wireless router by default. Note that end-to-end encryption does not protect the IP headers that can be considered as part of the privacy. For example, frequent accesses to certain bank web site imply the user's personal bank account. Similar to the incoming traffic control at the wireless router, the scheduler's control over outgoing traffic can also be customized by the user.

3.4 High-performance flow scheduling

Because the residential user typically cannot change any network component outside the home, PERM schedules traffic at the flow level for immediate deployability. To address the lack of flexibility of flow scheduling, PERM exploits its proximity to the end-user, and builds in predictive pre-probing and scheduling based on automated on-line learning of the user's networking traffic. PERM first infers the set of remote IPs with which the user will most likely initiate connections in the short term, and estimates the traffic volume involved with each flow. PERM then pre-probes these remote addresses to measure the quality of each available Internet connection. Finally PERM schedules flows based on the above knowledge to optimize performance metrics, depending on the type of the traffic. We summarize PERM's high-performance flow scheduling in this section. Interested readers are referred our previous work [33] for the complete description and analysis of the scheduling algorithms.

3.4.1 User traffic characterization

Users perceive the network performance differently for different applications. It is therefore important for the scheduler to recognize the types of traffic so that it can optimize relevant metrics accordingly. Although PERM supports an option for receiving application feedback, it is not resonable to expect existing applications to be recompiled in order to specify flow characteristics. Instead the framework infers the type of flow being scheduled and reacts accordingly. The first hints are the port number used in the connection. Well known ports such as port 80 (html), 20 and 21 (ftp), 22 (ssh), 25 (smtp), 110 (pop3), etc. can be used to immediately make a strong inference about the type of traffic. The type of traffic can also be inferred based on the flow characteristics. Using flow volume (total bandwidth), duration, and the transport protocol we can make reasonable inferences of the traffic type. Small volume flows (web pages, email download, ssh session) are most affected by link latency since they typically do not fill the pipe. However, large volume flows like bulk transfers, are more dependent on available bandwidth.

Unfortunately the flow volume is not known for certain until the flow finishes. The PERM scheduler predicts the volume of the next connection to a specific remote IP address using LMMSE (linear minimum mean square error), and achieves an average prediction error of less than 20% for flows with more than 20KB volume, when evaluated against the four-month residential building traces collected at Dartmouth campus [3].

3.4.2 Predictive pre-probing

For optimized performance it will be beneficiary for the PERM scheduler to have the round-trip-time, jitter, and available bandwidth over different links to a remote Internet address. Probing a destination after the connection request is received incurs too much latency. On the other hand, pre-probing even a large set of hosts from a residential host is not a viable option. PERM solves this problem through predictive pre-probing. That is, PERM predicts the remote IP addresses that a user is likely to visit and measures link metrics to the given destination.

Our prediction is made based on the intuition that certain destinations are associated. By building an association table between addresses the prober can selectively choose likely upcoming addresses to measure in the backgound. Our predictive prober had a 92% hit ratio for web flows and an overall 53%. In comparison a history-based prober only achieves a 53% web and 28% overall hit ratio.

Given the set of likely upcoming remote IP addresses, round-trip-time and jitter can be easily probed. The other important parameter for scheduling flows is available bandwidth. There is much ongoing research into accurately measuring available bandwidth (pathload, ProbeGap, etc.). However, these approaches require either external support or are too expensive for our scenario. Fortunately the PERM scheduler does not demand a highly accurate bandwidth estimate so PERM simplifies the measurement with a less accurate approach. Our measurement to the top 500 websites in Alexa.com shows that for 86% of the websites our DSL link is the bottleneck and for 77% our Cable is the bottleneck. PERM therefore estimates the access link capacity by periodically sampling the channel and measuring the peak activity. The accuracty of this method fluctuates depending on the amount of cross traffic at the wireless router. However, it is easy to measure passively without probing the network and it produces results good enough in our experiments.

3.4.3 Predictive scheduling

Given the combination of prediction, active probing, and passive monitoring a PERM scheduler develops a complete picture of the state of the flow scheduling. We adopt a simple greedy scheduling algorithm given above link and flow classification. The scheduler works by attempting to maximize the utility function that matches the current flow. For web traffic (small volume TCP flows) the scheduler uses round trip times to select the best link. For large volume flows the scheduler decides based on the measured available

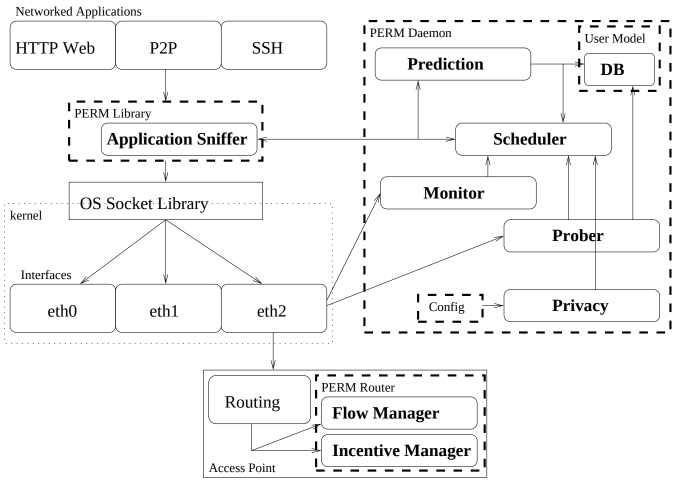

Figure 3: PERM Prototype Framework

bandwidth. Real-time traffic is scheduled based on jitter, and interactive applications are scheduled based on delay.

4. IMPLEMENTATION

The implementation of PERM is composed of two parts: the incentive and flow manager at the wireless router and the flow scheduler at the client, as illustrated in Figure 3. We have implemented the PERM incentive and flow manager for the open-source Linksys WRT54G 802.11g wireless router, and the PERM scheduler for Linux client. In total our implementation has around 10K lines of C codes.

4.1 PERM Wireless Router

We implement PERM's incentive and flow manager in the open-source Linksys WRT54G, a popular, highly-customizable home wireless router in the market. It has four 10/100 Ethernet ports, one WAN port and 802.11g wireless interface. The version we used to build our prototype is equipped with 16MB RAM, 4MB flash and 200MHz MIPS CPU. For operating system we choose OpenWRT Linux [5], one of many different Linux distributions for WRT54G developed by the open-source communities.

We build PERM's authentication on top of existing 802.11 industry-standard WPA (Wi-Fi Protected Access). We operate WPA in the Enterprise mode, since we expect PERM systems to discover each other automatically and no preshared keys can be configured. We configure an on-site FreeRADIUS server in each PERM wireless router. A PERM user and a PERM wireless router mutually authenticate each other, for credit maintenance and security privacy enforcement, through extensible authentication protocol (EAP). Specifically, we use EAP-TLS that identifies entities using digital certificates, compliant with our certified identity control as described in Section 3.2. We generate our own Certificate Authority using `openssl`. Two certificates are issued per user. One for the incentive and flow manager at the wireless router, and the other is for the scheduler shared among all clients that belong to the same user. Both certificates bear the same `Common Name` field so that they are easily recognized.

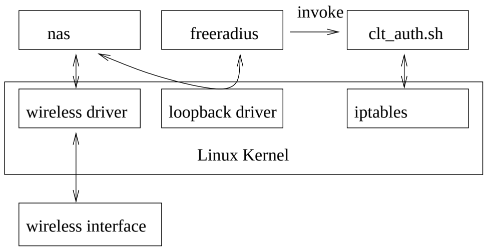

Figure 4: Architecture of Access Points

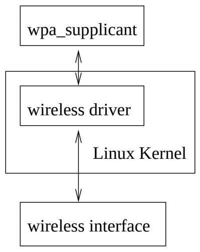

Figure 5: Architecture of clients

Figure 4 shows the architecture of PERM Linksys WRT54G wireless router. "`nas`" is the Broadcom's proprietary binary tool that sets up dynamic encryption (WEP/WPA) on the wireless router. When the wireless router boots, `nas` is executed with parameters related WPA such as authentication mode (wpa, wpa2, psk, etc.), encryption mode (TKIP, AES, etc.), RADIUS server IP/port, and so on. `nas` and FreeRADIUS communicates through the loopback interface. The architecture of normal linux-base WPA clients, depicted in Figure 5, is relatively simple. "`wpa_supplicant`" is responsible for WPA mutual authentication with PERM certificates.

To enforce the credit-based traffic shaping as presented in Section 3.2, we apply HFSC (hierarchical fair service curve) [31] class-based traffic scheduler that comes with the Linux 2.6 kernel. It supports bandwidth allocation for real time traffic as well as regular class prioritization. We define three classes for home client traffic, credited foreign client traffic, and uncredited foreign client traffic respectively, with decreasing priorities. A rate limit is specified for the traffic initiated by credited foreign clients. It is enforced to cap the total amount of traffic of credited foreign clients, unless the home clients are all idle. A relatively lower rate limit is specified for the traffic initiated by uncredited foreign clients. No further fairness is enforced among flows of the same class.

After a neighbor client authenticates with a user's wireless router, Linux `iptables` are configured to restrict access to certain IP addresses. Since WPA is a link-layer security mechanism it does not prevent an authenticated user from changing their IP addresses. We therefore identify clients using their MAC address in the `iptables`.

An image file of 3.2M bytes contains all software components at the wireless router. A user will need to download the image file, with the authentication credentials embedded, and update his Linksys WRT54G wireless router.

Function	Original Linux	PERM Library	Increase
`socket()`	6 usec	58 usec	52 usec
`setsockopt()`	2 usec	2 usec	0 usec
`getsockopt()`	2 usec	2 usec	0 usec
`bind()`	11 usec	34 usec	23 usec
`connect()`	34 usec	11855 usec	11821 usec
`send()`	13 usec	15 usec	2 usec
`recv()`	7 usec	12 usec	5 usec
`close()`	12 usec	54 usec	32 usec

Table 1: Average elapse time, in microseconds, of network system calls. Times of the normal Linux system calls are listed on the left. Times of the calls in the PERM framework are in the middle. The increase is given in the final column.

4.2 PERM Library

The purpose of the PERM library is to unobtrusively collect application level information and to bind application sockets to the appropriate interface. To stay transparent, PERM library sits in between the applications and the socket layer, and intercepts system calls from the applications. We take advantage of Linux's support for dynamic library loading to transparently interact with existing applications. PERM library intercepts application calls to the Linux socket wireless routerI. An application's socket function calls are dynamically linked to the PERM library implementations through `LD_PRELOAD`. The library in turn interacts with the original `libc` functions and PERM daemon to perform flow scheduling. The PERM library collects information about new flows, final flow volume measurements, and instances of application `binds`. When the application attempts to connect a socket, the library retrieves from the backend the interface the socket should use. The backend uses the previously reported information to select the best interface for the flow. After the library learns which interface to use, it binds the application socket to that interface.

By modifying the original `libc` socket library calls some extra processing overhead is introduced. We measured the overhead increase for each of the intercepted function calls in a Pentium III client. The results are shown in Table 4.1. For most of the functions, the change is negligible. However, there is a major increase in the `connect` call as well as significant increases in `socket`, `bind` and `close`. For the latter three the increase is approximately the same and can be explained by the extra RPC call to the PERM daemon needed to transfer application information. The modifications to `connect` are more complex and include two RPC calls, flow scheduling, and binding of the socket. After further measurement it was shown that the extra overhead is due to the scheduling algorithm runtime. We believe that it is possible to further decrease this overhead by optimizing the flow scheduler implementation. Note that although the increase of the elapse time for `connect` is significant (to around 11.9 ms), it still allows around 84 `connect`'s per second, which is unlikely to be a bottleneck at residential broadband connection speed.

The PERM library also exports an extended socket API by which an application can provide specific flow information. We stress that because of our predictive scheduling an application can receive the benefits of the PERM framework without using this special API. However, scheduling can be improved with the extra information that an application might provide. Flow attributes are set via the `setsockopt` call. The PERM library defines a new socket level `SOL_PERM` which distinguishes PERM options from other socket options. If the socket level in the `setsockopt` call is set to `SOL_PERM` then PERM processes the options. Otherwise, the call is forwarded to `libc`. Currently PERM allows an application to specify four flow characteristics: volume, delay, jitter, and priority. The first, volume, is the number of bytes the application plans to transmit/receive on the connection. The other three, delay, jitter, and priority are requirements that the application desires to have. Each of the latter attributes can be given a value of either HIGH, MEDIUM, or LOW, as well as hard values.

4.3 PERM Daemon

The PERM daemon runs continuously in the user space on the client system, accepting connections from the PERM library via a UNIX local socket. The daemon contains the prediction, probing, monitoring, privacy and scheduling components. These components interact primarily through a database containing information about different IP addresses and through a list of available interfaces including statistics for each interface.

Prediction The PERM prediction mechanisms are based on historical information. The volume predictor depends on the previous flow volumes for a given address and the IP predictor depends on the history of previously visited destinations. This information is stored in the IP database. Each visited IP address has an entry in the database. Each record has a volume history of ten items, a list of the ten IP addresses most frequently visited after the given address and a cache of recently visited addresses not in the top ten. In addition there is a timestamp field for each address.

Once the PERM library detects that a flow has been closed it reports the total final volume of the flow to the daemon. The volume history is updated with the new information and stored in the database. The history is then later used at scheduling time to predict the expected volume for a flow.

After a new connection is established, the destination address is tracked for a period of time. For each subsequent connection the list of frequently visited neighbors for the original connection is updated. Each entry in the favorites list has an associated count for the frequency of visits, and a timestamp for the last visit. A destination is removed from the favorites list if its count is too low and its timestamp is too old. A cache is used to allow new entries to work their way into the favorites list. The favorites list for each address is used to decide targets to probe in the near future. When a connection to an address is created, the favorites list of that address are loaded into the prober's target list.

Probing The prober measures round trip times and jitter for each host in the target list. In order to probe destinations a series of <SYN,ACK> packets are sent one after another. Most TCP implementations respond to the <SYN,ACK> with an <RST> packet to break the invalid sequence. The time elapsed between sending and receiving a packet is used to calculate round trip time. Packets are padded to 1400 bytes (slightly under the MTU for both DSL and Cable) to ensure an accurate measurement of what application packets are likely to experience. Jitter is updated by comparing the round trip times from each of the received packets in the sequence. The results of the measurement are also stored in the IP database record of the target IP address for each link. The scheduler later queries the data when choosing a link.

Monitoring Each link is also passively monitored by periodically sampling active traffic on the link. The monitor samples the channel every five seconds to reduce overhead. The monitor measures current activity on the channel as well as storing the peak throughput captured. The measurements are stored as part of a list of available interfaces. Upon initialization a list is generated containing entries for each available physical interface. The monitor uses the list to determine which links to monitor. Also, separate statistics are kept for each link as part of the data structure.

Privacy The daemon reads a configuration file which contains the blacklist of restricted IP addresses, DNS names, and protocol port numbers. Also in the configuration file is the "home" interface over which the end-host is connected to the home wireless router. The scheduler uses the blacklist to ensure that private flows are only scheduled on the home wireless router.

Scheduling When a socket is connected the scheduler is invoked. First the scheduler classifies the type of flow based on the estimated flow volume. We apply a threshold of 20KB to classify low-volume and high-volume TCP flows, based on our analysis of the web flow traces [3]. For UDP, low-volume flows are considered interactive traffic while higher volume is considered real time. We used 50 KB as the cut off, which is lower than most media streaming applications and also allow some leeway for the interactive applications. For UDP flows because the flows are usually long-lived, the library reports the volume/time of the flow.

After the flow has been classified the scheduler analyzes each available link by looking at the round trip time to the given destination on that link (as measured by the prober), the available bandwidth (as measured by the monitor) and the permissions as determined by the security policy (the blacklist). Then the flow is scheduled using the afore mentioned algorithms.

5. PERFORMANCE EVALUATION

We deploy PERM on a small testbed in one author's home. The testbed consists of two Pentium-III laptop clients running the 2.6.5 Linux kernel, and two LinkSys WRT54G wireless routers. Each laptop client is equipped with an internal Intel 802.11a/b/g wireless interface, as well as an external NetGear 802.11g USB card. The two wireless routers are connected to two different ISPs, one on a DSL and the other on a Cable provider. The maximum achieved throughputs on the Cable connection were 5.5Mbps downlink and 440Kbps uplink. The DSL achieved 1.32Mbps downlink and 400Kbps uplink.

In this section, we first evaluate the performance gain for a single client when the other client is inactive, as one form of statistical gain in resource sharing. We then show how incentive and flow manager, implemented at the PERM wireless router, controls the bandwidth sharing when both clients are active at the same time. Finally we compare PERM flow scheduling with known schedulers for various types of flows.

5.1 Single-client performance gain

PERM improves residential users' Internet connectivity by pooling all last-mile Internet connectivity in range. To quantify the benefits we first measure the reliability improvement in our testbed. We then compare the performance of each type of flows, with and without PERM.

Reliability
One obvious benefit of the PERM framework is that it immunizes residential users from individual local ISP service outages. Outages occur frequently in residential broadband services ranging in time from hours to days. In fact, our Cable network experienced two outages in the last week alone, each lasting for several hours. Because of its probing capabilities PERM can detect the down link and schedule traffic to other connections. Other causes of connection failures may come from the Internet such as long BGP convergence latency and link and router failures. Many of these failures can also be avoided with PERM, as long as the connections along different access networks traverse different routes.

We traced the paths through the local Cable and DSL ISPs to five hundred web servers chosen from the list of most frequently visited web sites given at Alexa.com. This list contains hosts in nearly every continent and many countries. For each host z we used the traceroute tool to find the routes over each link a. For each path to a single host we found the first node where the two paths intersected, host s. We then calculated the length of the path from s to z and likewise end-host a to s. To measure the overlap we calculated the fraction of the hops along the path s-z compared to the total hops a-z. On average, paths had only 33% overlap in our testbed.

The low overlap only helps reliability if the failures are neither at the remote server nor in the last hops s-z. Therefore we repeatedly probed the above hosts on each access network over a period of one week to determine the potential improvement offered through PERM. Around 32% of the probes failed on one of the access networks. However, in only 4.8% of the cases did the probe fail for both connections. These results suggest that PERM can bring ∼27% improvement in reliability.

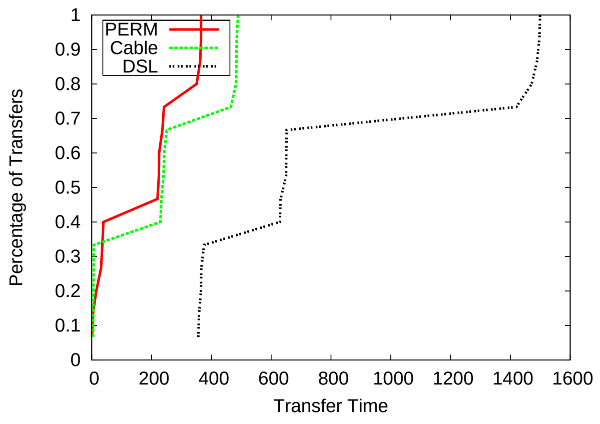

Figure 6: Completion times for bulk transfers

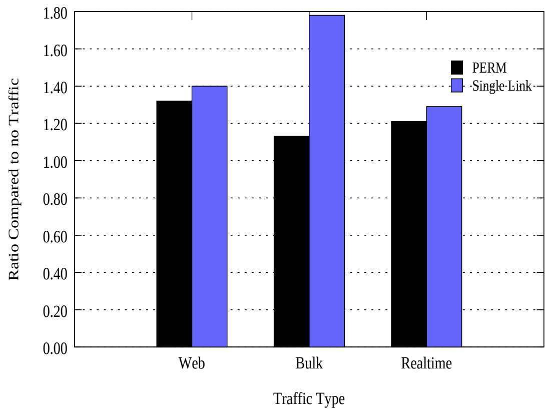

Figure 7: Increase in jitter with and without PERM

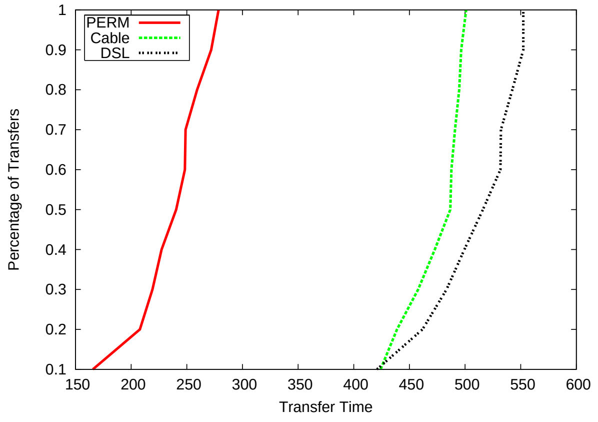

Figure 8: Completion time of upload flows

Web surfing
To evaluate the performance for web surfing we measured the completion time to fetch the index pages of popular web sites. We reused the list of top 500 web sites listed in alexa.com as our benchmark sites. Each index page was downloaded using the `wget` tool while using PERM, with just Cable and just DSL. The length of the entire transfer was recorded for each instance and compared. We ran the trials at similar times of the day.

We examined the average download times for each site. For some sites the downloads failed on every attempt so we removed those flows from the list. In the ideal case PERM's prediction mechanisms would always lead to selecting the interface with lowest round trip times and thus lowest transfer times. Our results show that for 46% of the flows PERM out performed the Cable and for 45% of the flows PERM out performed the DSL. For an additional 28%/38% of the flows PERM download times were within 1% of the respective links. For the remaining around 10% of flows the PERM scheduler performed worse than both of the other links. Likely causes include extra latency due to the interference from the prober and the stale estimate of round trip time.

Bulk download
To measure bulk flow performance we initiated several simultaneous downloads. Because a single flow sometimes failed to saturate the single link, we initiated three simultaneous flows to guarantee that the links would be saturated. The files downloaded were around 2MB, 10MB and 29MB, hosted on different web servers. We iterated the procedure 10 times and measured the transfer time of each individual flow.

The average times are shown in Figure 6. There are three separate "steps" in the CDF which correspond to the three different sized files. The average flow completion time for PERM was 21% lower than Cable and 81.3% lower than DSL. Note that although PERM does not outperform single link cases for every individual flow, the aggregate transfer time is minimized. Considering the 4:1 downlink capacity ratio between Cable and DSL in our testbed, the performance gain of PERM is close to optimum.

Real-time flows
To evaluate PERM's performance gain with real-time flows we launched a RealVideo stream on our client. With no competing traffic on just the Cable or the DSL link the video playback was smooth. As additional traffic was added to the line the playback became noticeably choppy and pixelated. Using PERM the performance was noticeably better, although there was some slight distortion under the bulk transfer cross traffic.

To quantify the performance of real-time flows under PERM we added multiple cross traffic flows of different types. The increase in jitter of the REalVideo stream under each type of cross traffic is shown in Figure 7. The figure shows the ratio of jitter with cross traffic compared to jitter without cross traffic. The strength of PERM is that it can isolate the real-time flow from competing traffic by scheduling them onto different connections. The benefits are especially evident as PERM kept the jitter one third lower than the jitter on a single link under bulk transfer traffic.

Uploads
Because of the popularity of p2p applications and swarming downloads, the utilization of the access link is becoming much more symmetric than before. PERM can schedule outgoing flows by controlling the interface on which an application binds it's socket. Figure 8 shows the improvement

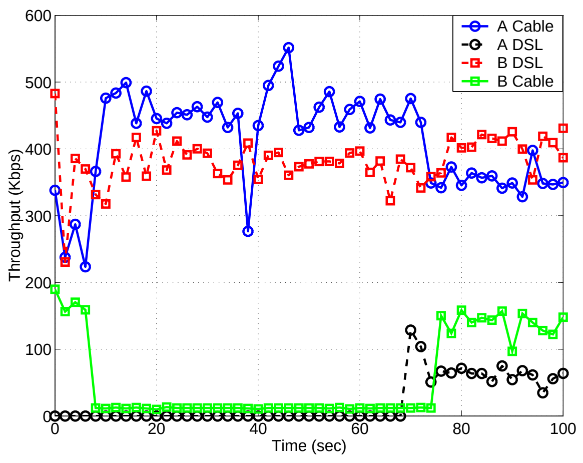

Figure 9: Two active clients

in average upload times using PERM. For the experiment we started two web servers on our client. We then initiated downloads from different remote hosts. Clearly PERM cuts the uploading time by half for all cases.

5.2 Incentive and flow management

We then show how incentive and flow manager at a PERM wireless router controls the bandwidth sharing and maintains credits. To this end we associate one client laptop and wireless router in our testbed to user A (identified by the certified name "User A"), and the other client laptop and wireless router to user B (identified by the certified name "User B"). We further assume user A subscribes to our Cable ISP and user B subscribes to our DSL ISP.

At the Cable wireless router, or the home wireless router for user A, we configure the rate limit of credited foreign clients to 200Kbps, equivalent to the aggregate speed of three fast dial-ups. We further configure the rate limit for uncredited foreign clients to 10Kbps and the initial credits for a foreign client to 150KB, representing the scenario when user A is charged by a flat broadband subscription fee. At the DSL wireless router, or the home wireless router for user B, we configure the rate limit of credited foreign clients to 100Kbps. We then set both the rate limit for uncredited foreign clients and the initial credit for a new foreign clients to zero, representing a cautious user B charged by usage.

In our experiment both client A and client B constantly run multiple `wget` instances, downloading websites randomly chosen from alexa.com top 500. The interactions between user A and B over a 100-second time span is shown in Figure 9. From the figure we can see that for the first 68 seconds user A serves client B at 10Kbps at A's Cable wireless router and accumulates credits slowly. Starting from the 68th second client A accumulates enough credits, and starts to be served by user B's DSL wireless router. Meanwhile, user B starts to earn credits by routing client A's traffic. Finally both client A and B accumulate enough credits and start to serve each other, at the rate pre-configured for credited foreign clients.

5.3 PERM scheduling

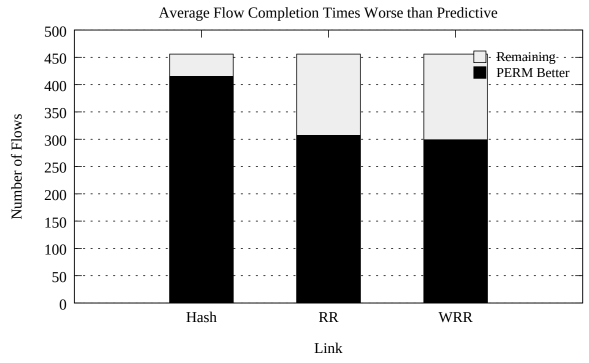

Figure 10: Number of flows from each scheduler which had longer completion times than the predictive scheduler

We finally investigate the effectiveness of the predictive scheduler compared to other well known schedulers. We implemented three schedulers for comparison: 1) a hash based scheduler that hashes the destination address to a particular interface, similar to the one popular in enterprise traffic load-balancing, 2) a round robin scheduler which iterates through all of the available interfaces, and 3) a weighted round robin scheduler that weighs each interface according to the achievable bandwidth of the interface.

IP prediction

Firstly we evaluated PERM's ability to predict upcoming IP addresses. This prediction is necessary in order for the selective probing to be useful in terms of accurate round trip times and jitter. We installed the framework on the author's portable computer and observed PERM's accuracy during normal Internet usage, including web browsing, email, ssh, occasional streaming news video, etc. To measure the effectiveness of the predictor we counted the number of times a new connection destination was in the list of probe targets. The predictor achieved a surprisingly high hit ratio of 83.04%. The result also confirms the fact that destination IP addresses are predictable and often visited in conjunction.

Short flow

Because PERM bases it's scheduling of web surfing and email flows on destination round trip time, it can outperform the other schedulers as long as PERM does not schedule the wrong interface because of stale or incorrect statistics. We reran the web download experiment from before for each of the schedulers. PERM indeed outperformed each of them. Figure 10 shows the number of flows for which the average download times using the predictive scheduler were better than the averages from the other schedulers. The predictive scheduler had the shortest download times for well over half of the flows in each case. For other cases the differences are all within 1% of each other.

Bulk transfer

We ran the bulk transfer experiment using each of the schedulers and measured the utilization of each link. The link utilization over time and the link busy time are shown in Figure 11 and Table 2 respectively. From Figure 11 we can see that the predictive scheduler maintains a higher utilization over both links for a longer period of time. The hash

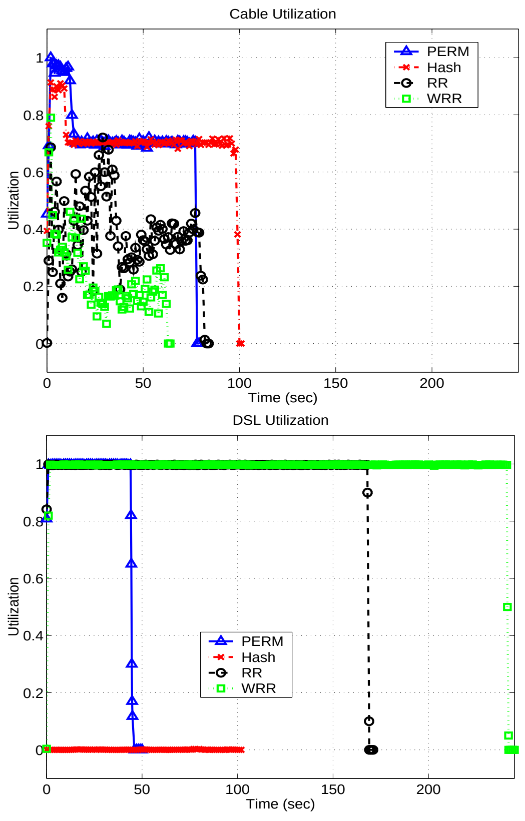

Figure 11: Link utilization. Upper: Cable. Lower: DSL

Scheduler	Cable	DSL
PERM Predictive	77	45
Hash	99	0
Round Robin	82	168
Weighted Round Robin	62	241

Table 2: Link busy time (seconds)

based scheduler used one link nearly exclusively reverting to a single link case, because the files were transferred from a small number of five web hosts. Weighted round robin and regular round robin too quickly drop off on one of the links, leading to longer download times on the other link. The link busy time in Table 2 further shows the predictive scheduler's ability to better balance the loads, although not perfect since PERM schedules traffic at the flow level. These results demonstrate the importance of flow volume prediction and available bandwidth estimation when scheduling bulk transfer.

Real-time flows

Finally we compared the ability of the different schedulers to isolate real-time flows. Figure 12 shows the jitter increases as percentages of the base jitter with no cross traffic, when multiple cross traffic flows of the same type are initiated simultaneously. By scheduling real-time flows on the same link the PERM scheduler indeed limits the jitter by at least

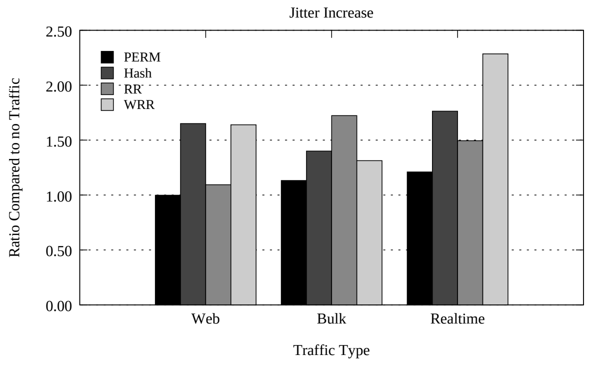

Figure 12: Jitter increases with Web surfing and bulk transfer cross traffic

10% and up to 47% lower than other schedulers which are oblivious of the flow types.

6. DISCUSSION

One major concern on the wide adoption of PERM is that PERM potentially increases the load on the local/regional ISP networks. For similar reasons very few ISPs allow a user to share the Internet connections, except for a few such as Speakeasy, Stephouse Networks, and EasyStreet (see [4] for a more complete list). We believe that the adoption of PERM will benefit not only a PERM user, but also broadband ISPs in that a user will need to subscribe to a broadband ISP before he is able to share with its neighbors via PERM. Furthermore, by distributing a user's bursty Internet traffic across multiple ISP networks, the instantaneous load of a single ISP network tends to be smoothed out over time. In other words, an ISP will join other ISPs to serve a larger Internet user pool, achieving an overall higher statistical multiplexing gain. With PERM's default symmetric sharing policy the load on any individual ISP network increases only if the total amount of traffic of all users in a residential area increases, which appeals to ISPs that charges by usage. The full impact of such increase can be evaluated only after PERM is deployed in large scale.

7. CONCLUSION

Widespread deployment of home wireless networks in residential areas enables ubiquitous, untethered Internet connectivity. As high-speed wireless devices flourish and competition for last mile Internet access heats up, collaborative residential Internet access will become the norm instead of niche exceptions. We present PERM, a practical system that realizes the vision. PERM does not rely on any support outside of the target user. Instead, PERM uses predictive mechanisms to achieve high-performance flow scheduling. PERM also preserves a user's security and privacy, and enforces incentive management to encourage the participation of compliant users.

There exists a hot on-going debate on the social and legal impacts of using or stealing neighbors' wireless networks. Following the mainstream media a residential Internet user is often taught how to enable encryption and access control to prevent intruders and free-riders. In this paper we demonstrate that with fully automated management, even a normal residential user can practically benefit from opening up his 802.11 wireless router and sharing his broadband Internet connection. We are working on the extension of PERM from one-hop neighborhoods to a multihop community, and further beyond individual residential areas.

8. REFERENCES

[1] http://www.digitalhomecanada.com/index.php?option=com_content&task=view&id=551.

[2] April 2006 bandwidth report. http://www.websiteoptimization.com/bw/0604/.

[3] Dartmouth college campus-wide wireless network. http://www.cs.dartmouth.edu/ campus/.

[4] ISP Wireless Policies. http://wiki.personaltelco.net/index.cgi/IspWirelessPolicies/.

[5] OpenWRT Linux. http://www.openwrt.org/.

[6] Adiseshu, H., Parulkar, G., and Varghese, G. A reliable and scalable striping protocol. In *Proceedings of ACM SIGCOMM* (August 1996).

[7] Adya, A., Bahl, P., Padhye, J., Wolman, A., and Zhou, L. A multi-radio unification protocol for IEEE 802.11 wireless networks. Tech. Rep. MSR-TR-2003-44, Microsoft Research, July 2003.

[8] Akella, A., Maggsy, B., Seshan, S., Shaikh, A., and Sitaraman, R. A measurement-based analysis of multihoming. In *Proceedings of ACM SIGCOMM* (2003).

[9] Akella, A., Pang, J., Maggsy, B., Seshan, S., and Shaikh, A. A comparison of overlay routing and multihoming route control. In *Proceedings of ACM SIGCOMM* (2004).

[10] Akella, A., Seshan, S., and Shaikh, A. Multihoming performance benefits: An experimental evaluation of practical enterprise strategies. In *Proceedings of USENIX Annual Technical Conference* (2004).

[11] Andersen, D. G., Balakrishnan, H., Kaashoek, M. F., and Morris, R. Resilient overlay networks. In *Proceedings of ACM SOSP* (2001).

[12] Andersen, D. G., Balakrishnan, H., Kaashoek, M. F., and Rao, R. Improving web availability for clients with MONET. In *Proceedings of NSDI* (2005).

[13] Castineyra, I., Chiappa, N., and Steenstrup, M. The nimrod routing architecture. RFC 1992, August 1996.

[14] Chakravorty, R., Banerjee, S., Agarwal, S., and Pratt, I. MoB: A mobile bazaar for wide-area wireless services. In *Proceedings of ACM MobiCom* (2005).

[15] Chebrolu, K., and Rao, R. Communication using multiple wireless interfaces. In *Proceedings of IEEE WCNC* (March 2002).

[16] Cohen, B. Incentives build robustness in BitTorrent. http://www.bittorrent.com/bittorrentecon.pdf, 2003.

[17] Douceur, J. R. The sybil attack. In *Proceedings of International Workshop on Peer-to-Peer Systems* (2002).

[18] Goldenberg, D., Qiu, L., Xie, H., Yang, Y. R., and Zhang, Y. Optimizing cost and performance for multihoming. In *Proceedings of ACM SIGCOMM* (2004).

[19] Gummadi, K. P., Madhyastha, H., Gribble, S. D., Levy, H. M., and Wetherall, D. J. Improving the reliability of internet paths with one-hop source routing. In *Proceedings of USENIX OSDI* (2004).

[20] Guo, F., Chen, J., Li, W., and cker Chiueh, T. Experiences in building a multihoming load balancing system. In *Proceedings of IEEE INFOCOM* (2004).

[21] Hsieh, H.-Y., and Sivakumar, R. A transport layer approach for achieving aggregate bandwidths on multi-home mobile hosts. In *Proceedings of ACM MobiCom* (2002).

[22] Magalhaes, L., and Kravets, R. Transport level mechanisms for bandwidth aggregation on mobile hosts. In *Proceedings of IEEE ICNP* (2001).

[23] Maltz, D. A., and Bhagwat, P. MSOCKS: An architecture for transport layer mobility. In *Proceedings of IEEE INFOCOM* (March 1998).

[24] Moskowitz, R. Host identity payload and protocol. Internet-Draft, November 2001. Work in Progress.

[25] Rodriguez, P., Chakravorty, R., Chesterfield, J., Pratt, I., and Banerjee, S. MAR: A commuter router infrastructure for the mobile internet. In *Proceedings of ACM MobiSys* (2004).

[26] Savage, S., Anderson, T., Aggarwal, A., Becker, D., Cardwell, N., Collins, A., Hoffman, E., Snell, J., Vahdat, A., Voelker, G., and Zahorjan, J. Detour: Informed internet routing and transport. *IEEE Micro 19*, 1 (January/February 1999), 50–59.

[27] Sharma, P., Lee, S.-J., Brassil, J., and Shin, K. G. Handheld routers: Intelligent bandwidth aggregation for mobile collaborative communities. In *Proceedings of IEEE BROADNETS* (2004).

[28] Snoeren, A. Adaptive inverse multiplexing for widearea wireless networks. In *Proceedings of IEEE GLOBECOM* (1999).

[29] Snoeren, A., and Balakrishnan, H. An end-to-end approach to host mobility. In *Proceedings of ACM MobiCom* (2000).

[30] Stewart, R. RFC 2960: Stream control transmission protocol, October 2000.

[31] Stoica, I., Zhang, H., and Ng, T. S. E. A hierarchical fair service curve algorithm for link-sharing, real-time and priority services. *IEEE/ACM Transactions on Networking 8*, 2 (April 2000), 185–199.

[32] Tao, S., Xu, K., Xu, Y., Fei, T., Gao, L., Guerin, R., Kurose, J., Towsley, D., and Zhang, Z.-L. Exploring the performance benefits of end-to-end path switching. In *Proceedings of IEEE ICNP* (2004).

[33] Thompson, N., He, G., and Luo, H. Flow scheduling for end-host multihoming. In *Proceedings of IEEE INFOCOM* (2006).