Control and Understanding: Owning Your Home Network

R. Mortier, T. Rodden, T. Lodge, D. McAuley
School of Computer Science
University of Nottingham
Nottingham, UK
{rmm,tar,txl,drm}@cs.nott.ac.uk

C. Rotsos, A.W. Moore
Computer Laboratory
University of Cambridge
Cambridge, UK
{cr409,awm22}@cl.cam.ac.uk

A. Koliousis, J. Sventek
School of Computer Science
University of Glasgow
Glasgow, UK
{koliousa,joe}@dcs.gla.ac.uk

***Abstract*—Wireless home networks are increasingly deployed in people's homes worldwide. Unfortunately, home networks have evolved using protocols designed for backbone and enterprise networks, which are quite different in scale and character to home networks. We believe this evolution is at the heart of widely observed problems experienced by users managing and using their home networks. In this paper we investigate redesign of the home router to exploit the distinct social and physical characteristics of the home.**

We extract two key requirements from a range of ethnographic studies: users desire greater *understanding* of and *control* over their networks' behaviour. We present our design for a home router that focuses on monitoring and controlling network traffic flows, and so provides a platform for building user interfaces that satisfy these two user requirements. We describe and evaluate our prototype which uses NOX and OpenFlow to provide per-flow control, and a custom DHCP implementation to enable traffic isolation and accurate measurement from the IP layer. It also provides finer-grained per-flow control through interception of wireless association and DNS resolution. We evaluate the impact of these modifications, and thus the applicability of flow-based network management in the home.

I. Introduction

Consumer broadband Internet access is a critical component of the digital revolution in domestic settings: for example, Finland has made broadband access a legal right for all its citizens.[1] A growing number of services are now provided over the Internet, including government, entertainment, communications, retail and health. The growth of IP enabled devices over the last decade also means many households are now exploring the use of in-home wired and wireless networking, not only to allow multiple computers to share an Internet connection but also to enable local media sharing, gaming, and other applications. Despite the growth in Internet use and the explosion of interest in home networking, the opacity of networking technologies means that they remain extraordinarily difficult for people to install, manage, and use in their homes.

In this paper we explore issues surrounding *home networks*: highly heterogeneous edge networks, typically Internet-connected via a single broadband link, where non-expert network operators provide a wide range of services to a small set of users. While we focus on home networks in this paper, we note that many environments, e.g., small offices, coffee shops, hotels, exhibit similar characteristics and thus may benefit from similar approaches. Specifically, we discuss the technical design and prototype evaluation of our home router, having first derived two key requirements through empirical ethnographic study of several home networks in use. A user-focused evaluation of our prototype is on-going, and will be reported in future papers.

In this paper we present three distinct contributions:

- We elaborate on the nature of the problems and opportunities inherent to home networks (§II);
- We describe our home router and how its flow-based approach enables it to help improve the user experience (§III); and
- We present and evaluate protocol modifications that place the homeowner in more direct control of their network (§IV).

Finally, we present related work (§V) and conclude (§VI). Note that throughout this paper we refer to the individual managing the home network as the homeowner without loss of generality; clearly any suitably permitted member of the household, owner or not, may be able to exercise control based on specifics of the local context.

II. The Elephant in the Room

"The technical know-how required to set up a network and run music or video across cables or wi-fi, is 'the elephant in the room that no-one wants to talk about.'"[2]

In recent years many empirical studies have explored the clear mismatch between current networking technology and the needs of the domestic setting, in both the UK [1], [2], [3], [4], [5] and the US [6], [7], [8], [9], [10]. These studies present a weight of evidence that problems with home networking are not amenable to solution via a 'thin veneer' of user interface technology layered atop the existing architecture. Rather, they are *structural*, emerging from the mismatch between the stable 'end-to-end' nature of the Internet and the highly dynamic and evolving nature of domestic environments.

[1] http://www.bbc.co.uk/news/10461048

[2] http://news.bbc.co.uk/1/hi/technology/6949607.stm

A. Home Networks: Evolution?

Home networks use the same protocols, architectures, and tools developed for the Internet since the 1970s. Inherent to the Internet's 'end-to-end' architecture is the notion that the core is simple and stable, providing only a semantically neutral transport service. Its core protocols were designed for a certain context of *use* (assuming relatively trustworthy endpoints), made assumptions about *users* (skilled network and systems administrators both using connected hosts and running the network core), and tried to accomplish a set of *goals* (e.g., scalability to millions of nodes) that simply do not apply in a home network.

In fact, the home network is quite different in nature to both core and enterprise networks. Existing studies [1], [6], [10] suggest domestic networks tend to be relatively small in size with between 5 and 20 devices connected at a time. The infrastructure is predominately cooperatively self-managed by residents who are seldom expert in networking technology and, as this is not a professional activity, rarely motivated to become expert. A wide range of devices connect to the home network, including desktop PCs, games consoles, and a variety of mobile devices ranging from smartphones to digital cameras. Not only do these devices vary in capability, they are often owned and controlled by different household members.

To illustrate the situation we are addressing, consider the following two example scenarios, drawn from situations that emerged from our fieldwork to date, reported in more detail elsewhere [11]:

Negotiating acceptable use. *William and Mary have a spare room which they let to a lodger, Roberto. They are not heavy network users and so, although they have a wireless network installed, they pay only for the lowest tier of service and they allow Roberto to make use of it. The lowest tier of service comes under an acceptable use policy that applies a monthly bandwidth cap. Since Roberto arrived from Chile they have exceeded their monthly cap on several occasions, causing them some inconvenience. They presume it is Roberto's network use causing this, but are unsure and do not want to cause offence by accusing him without evidence.*

Welcome visitors, unwelcome laptops. *Steve visits his friends Mike and Elisabeth for the weekend and brings his laptop and smartphone. Mike has installed several wireless access points throughout his home and has secured the network using MAC address filtering in addition to WPA2. To access the network, Steve must not only enter the WPA2 passphrase, but must also obtain the MAC addresses of his devices for Mike to enter on each wireless access point. Steve apologises for the trouble this would cause and, rather than be a problem to his hosts, suggests he reads his email at a local cafe.*

In such ways, simple domestic activities have deep implications for infrastructures that generate prohibitive technical overheads. In the first scenario, the problem is simply that the network's behaviour is opaque and difficult for normal users to inspect; in the second, the problems arise from the need to control access to the network and the technology details exposed by current mechanisms for doing so.

Home networks enable provision of a wide range of services, e.g., file stores, printers, shared Internet access, music distribution. The broad range of supported activities, often blending work and leisure, make network use very fluid. In turn, this makes it very hard to express explicitly *a priori* policies governing access control or resource management [1]. Indeed, fluidity of use is such that access control and policy may not even be consistent, as network management is contingent on the household's immediate needs and routines.

B. Home Networks: Revolution!

Simply creating a user interface layer for the existing network infrastructure will only reify existing problems. Rather, we need to investigate creation of new network architectures reflecting the socio-technical nature of the home by taking into account both human and technical considerations. For example, we may need to explore architectures that sacrifice scalability in favour of installability, evolvability, and maintainability.

To this end we exploit local characteristics of the home: devices are often collocated, are owned by family and friends who physically bring them into the home, and both devices and infrastructure are physically accessible. Essentially, the home's physical setting provides a significant source of heuristics we can understand, and offers a set of well understood practises that might be exploited in managing the infrastructure.

We exploit human understandings of the local network and the home to guide management of the supporting infrastructure [5] by focusing on the home router not only as the boundary point in an edge network but as a physical device which can be exploited as a point of management for the domestic infrastructure. Within our router, we focus on flow management for three reasons: (*i*) we do not require scalability to the same degree as the core network; (*ii*) doing so allows us to monitor traffic in a way that is more meaningful for users; and (*iii*) we can apply per-flow queuing mechanisms to control bandwidth consumption, a common request of users.

III. Reinventing the Home Router

Our home router is based on Linux 2.6 running on a micro-PC platform.[3] Wireless access point functionality is provided by the *hostapd* package. The software infrastructure on which we implement our home router, shown in Figure 1, consists of the Open vSwitch OpenFlow implementation, a NOX controller exporting a web service interface to control custom modules that monitor and manage DHCP and DNS traffic, plus the Homework Database [12] providing an integrated network monitoring facility. The proposed setup is similar to the standard ISP-provided home router.

We next describe the main software components upon which our router relies. Using this infrastructure, we provide a number of novel user interfaces, one of which we describe briefly below; details of the others are available elsewhere [13].

[3]Currently an Atom 1.6GHz eeePC 1000H netbook with 2GB of RAM running Ubuntu 10.04.

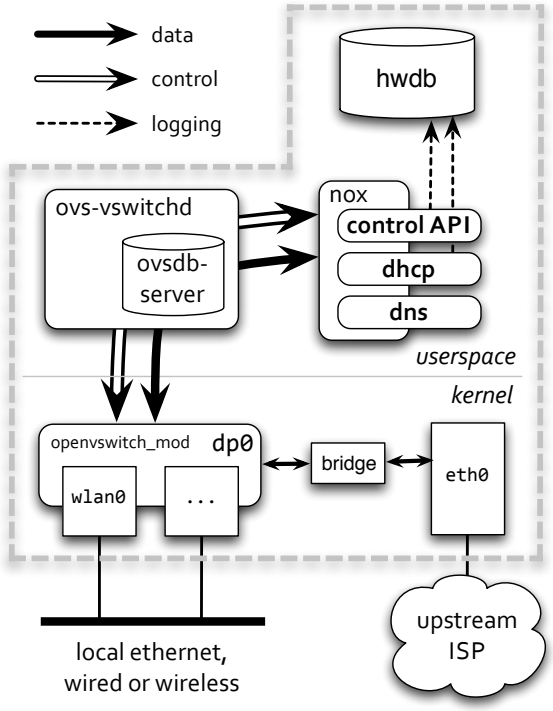

Fig. 1. Our home router architecture. Open vSwitch (*ovs**) and NOX manage the wireless interface. Three NOX modules provide a web services control API, a DHCP server with custom address allocation and lease management, and a DNS interceptor, all logging to the Homework Database (*hwdb*) (§IV).

Note that a key aspect of our approach is to avoid requiring installation of additional software on client devices: doing so is infeasible in a home context where so many different types of device, with such a high degree of variability in capabilities, remain in use over extended periods of time.

A. OpenFlow, Open vSwitch & NOX

OpenFlow is a switching standard [14] providing an open protocol for distributed control of the forwarding tables contained within Ethernet switches in a network. An OpenFlow *switch* has three parts: a *datapath*, a *secure channel* connecting the datapath to a controller, and the *OpenFlow protocol* used by the controller to talk to the switch.

Each datapath applies actions to flows detected on a physical interface, where *flow* is defined as a tuple of the primary packet header fields plus the physical port on which the flow is visible. Flow definition allows wildcarding of fields and specifically permits netmasks for IP addresses. Each flow can have a number of primitive actions applied; actions defined in the protocol permit full control over forwarding as well as modification of all fields of the flow tuple. The net effect is that applications can manage and control traffic according to their own definition of a network flow. Flow entries are installed by the controller when the switch notifies the controller of arrival of a packet from a new flow.

We provide OpenFlow support using Open vSwitch,[4] OpenFlow-enabled switching software that replaces the in-kernel Linux bridging functionality and is able to operate as a standard Ethernet switch as well as providing full support for the OpenFlow protocol. We use the NOX[5] controller as it provides a programmable platform abstracting OpenFlow interactions as events with associated callbacks, and exporting APIs for C++ and Python.

[4]http://openvswitch.org/

Method	Function
`permit/<eaddr>`	Permit access by specified client
`deny/<eaddr>`	Deny access by specified client
`status/[eaddr]`	List currently permitted clients, or status of specified client
`dhcp-status/`	List current MAC–IP mappings
`whitelist/<eaddr>`	Accept associations from client
`blacklist/<eaddr>`	Deny association to client
`blacklist-status/`	List currently blacklisted clients
`permit-dns/<e>/<d>`	Permit access to domain `d` by client `e`
`deny-dns/<e>/<d>`	Deny access to domain `d` by client `e`

TABLE I
WEB SERVICE API; PREFIX ALL METHODS `HTTPS://.../WS.V1/`. $<X>$ AND $[X]$ DENOTE REQUIRED AND OPTIONAL PARAMETERS.

Our router provides flow-level control and management of traffic via a single OpenFlow datapath managing the wireless interface of the platform.[6] We provide NOX modules that implement a custom DHCP server, control forwarding, control wireless association via filtering, and intercept DNS lookups. Control of these modules is provided via a simple web service (Table I). Traffic destined for the upstream link is forwarded by the datapath for local processing via the kernel bridge, with Linux's *iptables* IP Masquerading rules providing standard NAT functionality.[7]

B. The Homework Database

In addition to Open vSwitch and NOX we make use of the Homework Database, *hwdb*, an active, ephemeral streaming database [12]. The ephemeral component consists of a fixed-size memory buffer into which arriving tuples (events) are stored and linked into tables. The memory buffer is treated in a circular fashion, storing the most recently received events inserted by applications measuring some aspect of the system. The primary ordering of events is time of occurrence.

The database is queried via a variant of CQL [15] able to express both temporal and relational operations on data, allowing applications such as our user interfaces to periodically query the ephemeral component for either raw events or information derived from them. Applications need not be collocated on the router as *hwdb* provides a lightweight, UDP-based RPC system that supports one-outstanding-packet semantics for each connection, fragmentation and reassembly of large buffers, optimisation of ACKs for rapid request/response exchanges, and maintains liveness for long-running exchanges. Monitoring applications request events since a timestamp or during an interval defined by two timestamps. *hwdb* is active as

[5]http://noxrepo.org/

[6]Without loss of generality, our prototype has only a single wired interface so the only home-facing interface is its wireless interface; other home-facing interfaces would also become part of the OpenFlow datapath.

[7]While NAT functionality could be implemented as another NOX module, it seemed neither interesting nor necessary to do so.

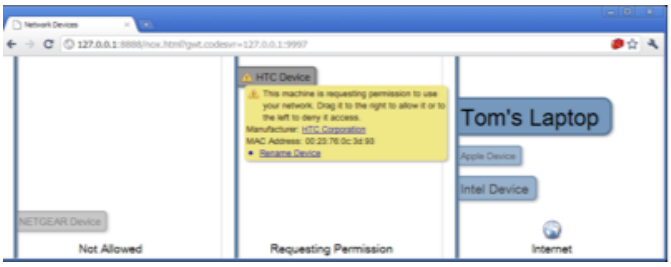

Fig. 2. The *Guest Board* control panel, showing an HTC device requesting connectivity.

applications may register interest in *future* behaviour patterns, triggering notification when such a pattern is detected by the database. The work described in this paper makes use of three tables: *Flows*, accounting traffic to each 5-tuple flow; *Links*, monitoring link-layer performance; and *Leases*, recording mappings assigned via DHCP.

C. The Guest Board

As an example of the kind of user interface we support, consider the second example scenario, of visitors with laptops (§II-A). The requirement elicited there was a need to enable easy admission of new devices to the home network. Our *Guest Board* interface depicted in Figure 2 exploits people's everyday understanding of control panels in their homes, e.g., heating or alarm panels, to provide users with a central point of awareness and control for the network. It runs on a dedicated touch screen in the home and we exploit this physical arrangement to provide a focal point for inhabitants to view current network status and to manage the network. It provides a real time display of the current status of the network, showing devices in different zones based on the state of their connectivity. The display dynamically maps key network characteristics of devices to features of their corresponding labels. Mappings in the current display are:

- Wireless signal strength is mapped to device label transparency.
- Device bandwidth use is proportional to its label size.
- Link-layer retransmissions show as red highlights on the device's label.

Devices in range appear on the screen in real-time, initially in the leftmost panel indicating they are within range of the home router but not connected. The central panel in the control displays machines actively seeking to associate to the access point: when devices unknown to the router issue DHCP requests, the router's DHCP server informs the guest board and a corresponding label appears in this portion of the display. If a user wishes to give permission for the machine to join the network they drag the label to the right panel; to deny access, they drag the label to the left panel.

The guest board provides both a central control point and, by drawing directly upon network information collected within our router, a network-centric view of the infrastructure. While this example describes a central control point in the home, the interface is implemented in HTML/CSS/Javascript allowing it to be displayed on a range of devices, currently under trial with users. The router's measurement and control APIs described above are also being used to build a wide range of other interfaces for use via smartphones, web browsers, and custom display hardware.

IV. Putting People in the Protocol

We use our home router to enable *ad hoc* management of the network by non-expert users via interfaces such as the Guest Board (Figure 2). This sort of control mechanism is a natural fit to the social context of the home, particularly the local negotiation over access and use that takes place in most homes. While we believe that this approach may be applicable to other protocols, e.g., NFS/SMB, LPD, in this section we demonstrate this approach via our implementation of a custom DHCP server and selective filters for wireless association and DNS that enable management of device connectivity on a per-device basis.

Specifically, we describe and evaluate how our router manages IP address allocation via DHCP, two protocol-specific (EAPOL and DNS) interventions it makes to provide finer-grained control over network use, and its forwarding path. We consider three primary axes: *heterogeneity* (does it still support a sufficiently rich mix of devices); *performance* (what is the impact on forwarding latency and throughput of our design and implementation decisions); and *scalability* (how many devices and flows can our prototype handle). In general we find that our home router has ample capacity to support observed traffic mixes, and shows every indication of being able to scale beyond the home context to other situations, e.g., small offices, hotels.

A. Address Management

DHCP [16] is a protocol for automatic host network configuration. It is based on a four-way broadcast handshake that allows hosts to discover and negotiate parameters of their connectivity with a server. As part of our design we extend the functionality of the protocol to achieve two goals. First, we enable the homeowner to control which devices are permitted to connect to the home network by interjecting in the protocol exchange on a case-by-case basis. We achieve this by manipulating the lease expiry time, allocating only a short lease (30s) until the homeowner has permitted the device to connect via a suitable user interface. The short leases ensure that clients will keep retrying until a decision is made; once a device is permitted to connect, we allocate a standard duration lease (1 hour).

Second, we ensure that all network traffic is visible to the home router and thus can be managed through the various user interfaces built against it. We do so by allocating each device to its own /30 IP subnet, forcing inter-device traffic to be IP routed via our home router. This requirement arises because wireless Ethernet is a broadcast medium so clients will ARP for destinations on the same IP subnet enabling direct link-layer communication. In such situations, the router becomes a link-layer device that simply schedules the medium and manages link-layer security – some wireless interfaces

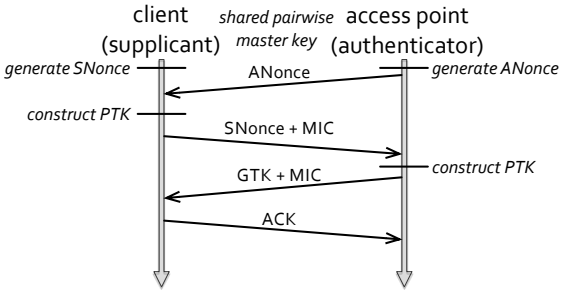

Fig. 3. 802.11i handshake, part of the association process. Note that MIC (Message Integrity Code) is an alternate term for Message Authentication Code, used in such contexts to avoid confusion with Media Access Control.

do not even make switched Ethernet frames available to the operating system, handling them entirely within the network card. Our custom DHCP server allocates /30 subnet to each host from 10.2.*.*/16 with standard address allocation within the /30 (i.e., when considering the host bits in the subnet, 00 maps to the network, 11 maps to subnet broadcast, 01 maps to the gateway and 10 maps to the client's interface itself). Thus, each local device needs to route traffic to any other local device thought the router, making traffic visible in the IP layer. This imposes only minor performance overheads, discussed in more detail in §IV-B.

Measuring the performance of our DHCP implementation, we found that per-request service latency scales linearly with the number of simultaneous requests, as expected. In a fairly extreme scenario, simultaneous arrival of 10 people each with 10 devices, we measured a median per-host service time of 0.7s.

B. Per-Protocol Intervention

Our current platform intervenes in two specific protocols providing greater control over access to the wireless network itself, and to Internet services more generally.

Our home router supports wireless Ethernet security via 802.11i with EAP-WPA2, depicted in Figure 3, using *hostapd*. In short, the client (*supplicant*) and our router (*authenticator*) negotiate two keys derived from the shared master key via a four-way handshake using the EAPOL protocol. The *Pairwise Transient Key* (PTK) is used to secure and authenticate communication between the client and the router; the *Group Transient Key* (GTK) is used by the router to broadcast/multicast traffic to all associated clients, and by the clients to decrypt that traffic. All non-broadcast communication between clients must therefore pass via the router at the link-layer for decryption with the source's PTK and re-encryption with the destination's PTK, although the IP routing layers are oblivious to this if the two clients are on the same IP subnet.[8]

[8]The 802.11i specification defines a general procedure whereby two clients negotiate a key for mutual communication (*Station-to-station Transient Key*, STK). However, the only use of this procedure in the specification is in *Direct Link Setup* (DLS) used in supporting 802.11e, quality-of-service. This can easily be blocked by the access point by dropping particular EAPOL frames, and in fact is not implemented in the *hostapd* code we use, so we do not consider it further.

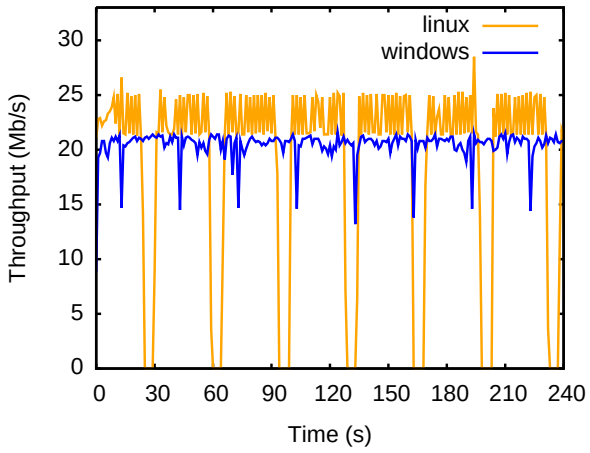

Fig. 4. Affect on TCP throughput from rekeying every 30s for Linux 2.6.35 using a Broadcom card with the *athk9* module; and Windows 7 using a proprietary Intel driver and card.

Periodically, a timeout event at the access point initiates rekeying of the PTK, visible to clients only as a momentary drop in performance rather than the interface itself going down. We use this to apply blacklisting of clients deemed malicious, such as a client that attempts to communicate directly (at the link-layer) with another, i.e., attempting to avoid their traffic being visible to our home router. We wait until the rekeying process begins and then decline to install the appropriate rule to allow it to complete for the client in question. This denies the client access even to link-layer connectivity, as they will simply revert to performing the four-way handshake required to obtain the PTK. This gives rise to a clear trade-off between security and performance: the shorter the rekeying interval, the quicker we can evict a malicious client but the greater the performance impact on compliant clients.

To quantify the impact of 802.11i rekeying Figure 4 shows the impact of setting the rekeying interval to 30s. Rekeying causes a periodic dip in throughput as the wireless Ethernet transparently buffers packets during rekeying before transmitting them as if nothing had happened. This shows the trade-off between performance and responsiveness of this approach: higher responsiveneness in detecting misbehaving clients will impose a small performance degradation network-wide. As a compromise, when a device is blacklisted, all of its traffic and subsequent rekeying exchanges are blocked. The device will be able to receive only broadcast traffic in the interim, due to the use of the GTK for such frames, until the AP initiate the negotiation of a new key.

We also intercept DNS to give fine-grained control over access to Internet services and websites. DNS requests are intercepted and dropped if the requesting device is not permitted to access that domain. Any traffic the router encounters that is not already permitted by an explicit OpenFlow flow entry has a reverse lookup performed on its destination address. If the resulting name is from a domain that the source device is not permitted to access, then a rule will be installed to drop related traffic. Performance is quite acceptable, as indicated

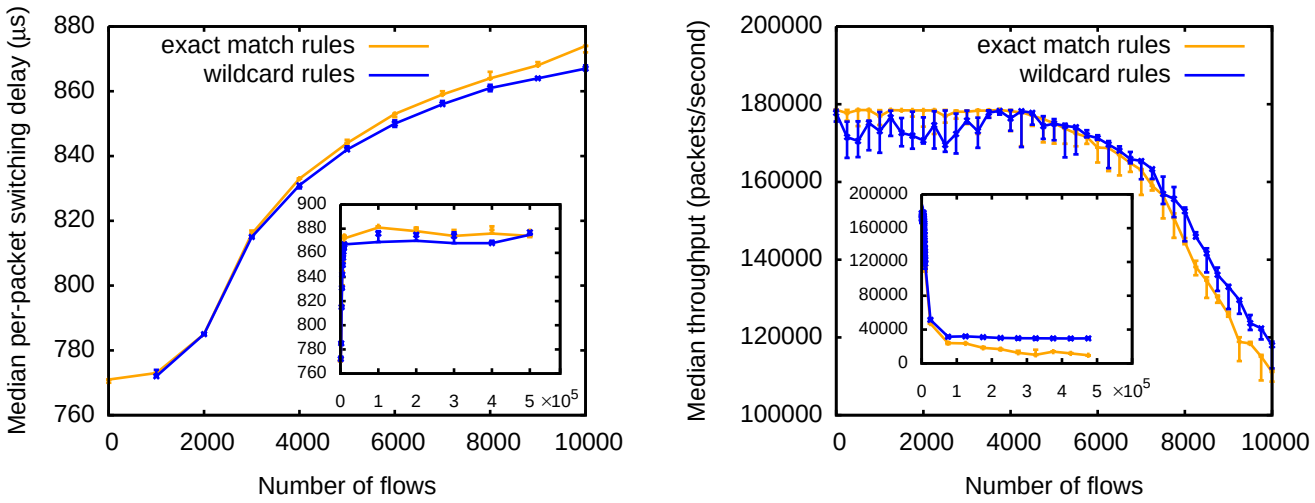

Fig. 5. Switching performance of Open vSwitch in our home router showing increasing per-packet latency (LHS) and decreasing packet throughput (RHS) with the number of flows. The inset graph extends the x-axis from 10,000 to 500,000.

by latency results in Figure 5: the extra latency overhead introduced by our router is negligible compared to the inherent latency of a lookup to a remote name server. Extending this fine-grained control requires more accurate identification of traffic to application, particularly for more complex network uses such as BitTorrent and Skype, and is a problem we are investigating in ongoing work.

C. Forwarding

Our router consists of a single Open vSwitch that manages interface *wlan0*. Open vSwitch is initialised with a set of flows that push DHCP/BOOTP and IGMP traffic to the controller for processing. Open vSwitch by default will also forward to the controller traffic not matched by any other installed flow. This traffic is handled as follows:

Non-IP traffic. The controller acts as a proxy ARP server, responding to ARP requests from clients. Misbehaving devices are blacklisted via a rule that drops their EAPOL [17] traffic thus preventing negotiation of session keys. Finally, other non-IP traffic has source and destination MAC addresses verified to ensure both are currently permitted. If so, the packet is forwarded up the stack if destined for the router, or to the destination otherwise. In either case, a suitable OpenFlow rule with a 30s idle timeout is also installed to shortcut future matching traffic.

Unicast IP traffic. First, a unicast packet is dropped if it does not pass all the following tests: (*i*) its source MAC address is permitted; (*ii*) its source IP address is in 10.2.*.*/16; and (*iii*) its source IP address matches that allocated by DHCP. For valid traffic destined to the Internet, a flow is inserted that forwards packets upstream via the bridge and IP masquerading. For local traffic a flow is installed to route traffic as an IP router, i.e. rewriting source and destination MAC addresses appropriately. All these rules are installed with 30s idle timeouts, ensuring that they are garbage collected if the flow goes idle for over 30s.

Broadcast and multicast IP traffic. Due to our address allocation policy, broadcast and multicast IP traffic requires special attention. Clients send such traffic with the Ethernet broadcast bit[9] set, normally causing the hardware to encrypt with the GTK rather than the PTK so all associated devices can receive and decrypt those frames directly. In our case, if the destination IP address is all-hosts broadcast, i.e., 255.255.255.255, the receiver will process the packet as normal. Similarly, if the destination IP address is an IP multicast address, i.e., drawn from 224.*.*.*/4, any host subscribed to that multicast group will receive and process the packet as normal. Finally, for local subnet broadcast the router will rebroadcast the packet, rewriting the destination IP address to 255.255.255.255. This action is required because the network stack of the hosts filters broadcast packets from different IP subnets.

To assess switching performance, we examine both latency and packet throughput as we increase the number of flows, N, from 1–500,000. Each test runs for two minutes, generating packets at line rate from a single source to N destinations each in its own 10.2.*.*/30 subnet. As these are stress tests we use large packets (500B) for the latency tests and minimal packets (70B) for the throughput tests, selecting destinations at random on a per-packet basis. Results are presented as the median of 5 independent runs with error bars giving the min and max values.

Figure 5 shows median per-packet switching delay and per-flow packet throughput using either exact-match rules or a single wildcard rule per host. Performance is quite acceptable with a maximum switching delay of 560μs and minimum throughput of 40,000 packets/second; initial deployment data suggests a working maximum of 3000 installed flows which would give around 160,000 packets/second throughput (small packets) and 500μs switching delay (large packets). Figure 6 shows that the Linux networking stack is quite capable of

[9]I.e., the most significant bit of the destination address

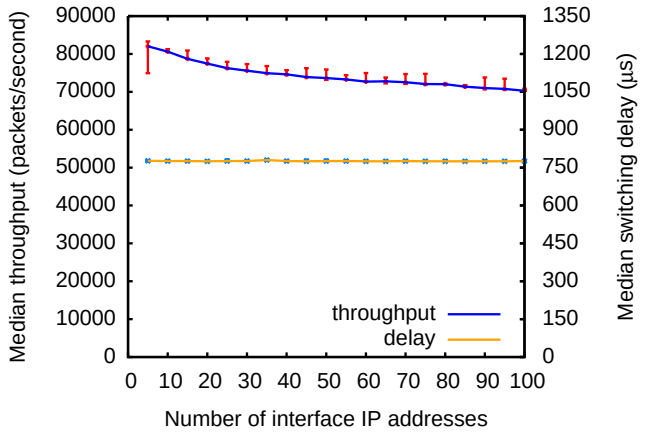

Fig. 6. Switching performance of Linux network stack with our address allocation policy. Throughput (left axis) shows a small linear decrease while switching delay (right axis) remains approximately constant as the number of interface addresses increases.

handling the unusual address allocation pattern resulting from the allocation of each wireless-connected device to a distinct subnet which requires the router's wireless interface to support an IP address per connected device.

D. Discussion

Our evaluation shows that Open vSwitch can handle orders of magnitude more rules than required by any reasonable home deployment. Nonetheless, to protect against possible denial-of-service attacks on the flow tables, whether intentional, accidental or malicious, our home router monitors the number of per-flow rules introduced for each host. If this exceeds a high threshold then the host has its per-flow rules replaced with a single per-host rule, while the router simultaneously invokes user interfaces to inform the homeowner of the device's odd behaviour.

The final aspect to our evaluation is compatibility: given that our router exercises protocols in somewhat unorthodox ways, how compatible is it with standard devices and other protocols? We consider compatibility along three separate dimensions: range of existing client devices; deployed protocols that rely on broadcast/multicast behaviours; and support for IPv6.

a) Devices: Although we exercise DHCP, DNS and EAPOL in unorthodox ways to control network access, behaviour follows the standards once a device is permitted access. To verify that our home router is indeed suitable for use in the home, we tested against a range of commercial wireless devices running a selection of operating systems.

Table II shows the observed behaviour of a number of common home-networked devices: in short, all devices operated as expected once permitted access. DNS interception was not explicitly tested since, as an inherently unreliable protocol, all networking stacks must handle the case that a lookup fails anyway. Most devices behaved acceptably when denied access via DHCP or EAPOL, although some user interface improvements could be made if the device were aware of the registration process. The social context of the home network means no problem was serious: in practice the user requesting access would be able to interact with the homeowner, enabling social negotiation to override any user interface confusion.

b) Broadcast protocols: A widely deployed set of protocols relying on broadcast and multicast behaviours are those for 'zero conf' functionality. The most popular are Apple's *Bonjour* protocol; *Avahi*, a Linux variant of Bonjour; Microsoft's *SSDP* protocol, now adopted by the UPnP forum; and Microsoft's *NetBIOS*.

Bonjour and Avahi both rely on periodic transmission of multicast DNS TXT records advertising device capabilities. SSDP is similar, but built around multicast HTTP requests and responses. We tested Bonjour specifically by setting up a Linux server using a Bonjour-enabled daemon to share files. We observed no problems with any clients discovering and accessing the server, so we conclude that Bonjour, Avahi and SSDP would all function as expected.

NetBIOS is somewhat different, using periodic network broadcasts to disseminate hosts' capabilities. In doing so we observed a known deficiency of NetBIOS: it cannot propagate information for a given workgroup between different subnets.[10] However this was easy to overcome: simply install a WINS server on the router and advertise it via DHCP to all hosts. Although something of a kludge, this works at little cost in comparison to the benefits of our approach. In particular, it does not increase the externally visible attack surface of the router and network since both Linux IPtables and Open vSwitch can be used to deny WINS access from outside the home network.

In general, it may seem that our address allocation policy introduces link-layer overhead by forcing all packets to be transmitted twice in sending them via the router. However this is not the case: due to use of 802.11i, unicast IP traffic between two local hosts must *already* be sent via the access point. As the source encrypts its frames with its PTK, the access point must decrypt and re-encrypt these frames with the destination's PTK in order that the destination can receive them. Multicast and all-hosts broadcast IP traffic is sent using the GTK, so can be received directly by all local hosts. Only directed broadcast IP traffic incurs overhead which though is a small proportion of the total traffic; data from a limited initial deployment (about one month in two homes) suggests that broadcast and multicast traffic combined accounts for less that 0.1% (packets and bytes) in both homes.

c) IPv6 support: IPv6 support is once more receiving attention due to recent exhaustion of the IPv4 address space. Although our current prototype does not support IPv6 due to limitations in the current Open vSwitch and NOX releases,[11] we briefly discuss how IPv6 would be supported on our platform. While these limitations prevent a full working implementation in our prototype, we have verified that the behaviour of DHCPv6 and pertinent ICMPv6 messages is as

[10] http://technet.microsoft.com/en-gb/library/bb726989.aspx

[11] OpenFlow aims to provide support in its 1.2 release of the protocol; NOX currently has no support for IPv6; and Open vSwitch only supports IPv6 as an application specific extension.

Device	Denied	Blacklisted
Android 2.x	Reports pages unavailable due to DNS.	Retries several times before backing off to the 3g data network.
iTouch/iPhone	Reports server not responding after delay based on configured DNS resolver timeout.	Requests new wireless password after 1–2 minutes.
OSX 10.6	Reports page not found based on configured DNS resolver timeout.	Requests new wireless password after 1–2 minutes.
Microsoft Windows XP	Silently fails due to DNS failure.	Silently disconnects from network after 4–5 minutes.
Microsoft Windows 7	Warns of partial connectivity.	Silently disconnects from network after 4–5 minutes.
Logitech Squeezebox	Reports unable to connect; allows server selection once permitted.	Flashes connection icon every minute as it attempts and fails to reconnect.
Nintendo Wii	Reports unable to reach server during "test" phase of connection.	Reports a network problem within 30s.
Nokia Symbian OS	Reports "can't access gateway" on web access.	Reports disconnected on first web access.

TABLE II
OBSERVED INTERACTIONS BETWEEN DEVICES AND OUR HOME ROUTER WHEN ATTEMPTING TO ACCESS THE NETWORK.

expected, so we do not believe there are any inherent problems in the approaches below.

Addition of IPv6 support affects the network layer only, requiring consideration of routing, translation between network and link layers, and address allocatiocn. Deployment of IPv6 has minimal impact on routing, limited to the need to support 128 bit addresses and removal, in many cases, of the need to perform NAT. Similarly, supporting translation to lower layer addresses equates to supporting ICMPv6 Neighbour Solicitation messages which perform equivalent function to ARP.

Address allocation is slightly more complex but still straightforward. IPv6 provides two address allocation mechanisms: *stateless* and *stateful*. The first allows a host to negotiate directly with the router using ICMPv6 Router Solicitation and Advertisement packets to obtain network details, IP netmask and MAC address. Unfortunately this process requires that the router advertises a 64 bit netmask, of which current plans allocate only one per household, with the result that all hosts would end up on the same subnet. The second builds on DHCPv6 where addresses are allocated from a central entity and may have arbitrary prefix length. This would enable our router to function in much the same manner as currently, although it would need to support the ICMPv6 Router Advertisement message so hosts could discover it as the router.

V. RELATED WORK

Many authors have argued that home networks should be treated differently to other IP-based networks. For example, Calvert *et al*. [18] make a case against application of the end-to-end principle in home networking. They argue that there are a number of key aspects peculiar to the home environment that the standard Internet protocols do not address. They derive a series of requirements for a home network architecture, and a design providing functions to fulfil these requirements. Many of their points, e.g., the "smart middle" design, resonate with our argument, and indeed, we believe our home router meets their requirements and provides the functions described.

Both before and after the general architectural arguments made above, a number of authors have proposed novel user interfaces to aid the homeowner in managing their network. GesturePen [19] uses line-of-sight radio interaction with purpose-built receiver tags to control network access; Network-in-a-Box [20], where infrared port alignment provides a physical metaphor for access, plugging in to security mechanisms such as EAP-TLS and RADIUS. They also describe an interesting "phone home" service via the Windows Remote Access and IPSec policy mechanisms that enables remote clients to connect back to appear as if inside the home network.

ICEBox [21] again concentrates on the problem of enabling the homeowner to correctly configure new devices when they are brought onto the network using a control panel approach similar to our Guest Board. They note a future version might well subsume the home router. Eden [22], by several of the same authors, follows up by replacing the home router. Their paper describes allowing per-flow traffic control but lacks technical details.

All these approaches primarily address the interaction design problem, focusing on user interface solutions to the problems of managing a home network. Most rely upon specialized hardware or software installation on client devices. In contrast, our home router does not require client modification as its protocol modifications are fully backward compatible with existing stacks. It thus supports a very wide range of devices while making possible greater control of connectivity.

Several authors have proposed solutions to the specific problem of lack of visibility into what the network and connected devices are doing. In this context, HostView [23] uses a client daemon to log when users experience network problems. Calvert *et al*. [24] present requirements for a general purpose "always on" local logging service, building a simple example using *tcpdump* running on a NOXBox,[12] dumping traffic into flat files. Both focus on network monitoring as a tool for troubleshooting. Finally, in presenting the Homework Database, Sventek *et al*. [12] describe it as a component in their home network information plane. They use a general-purpose policy engine to exercise control over the network by configuration of the router derived from the interaction of

[12]http://www.noxrepo.org/manual/noxbox.html

monitored data and policy.

We claim these monitoring systems do not generally take into account the specific challenges and opportunities inherent in the home network context. Our home router goes further in exploiting the home context via specific modifications to the normal behaviour of key protocols, and implementing a novel network control interface.

This class of argument, that the generic Internet protocols are not appropriate in a particular environment, has previously been made in the enterprise network space. Approaches such as Anemone [25], Ethane [26] and Network Exception Handlers [27] have all proposed systems that address the general problem of enterprise network management in different ways. They all make the argument that enterprise networks are basically different to the traditional Internet, presenting different problems and permitting different solutions. This resonates strongly with our claims that home networks should be treated differently, and in some ways with our approach of providing more intelligent centralised control. It should be noted however, that these enterprise network solutions are no more applicable to home networks than traditional Internet approaches were applicable in the enterprise!

Finally, looking back to 1984 and some of the original discussions of IP subnetting, Mogul [28] and Postel [29] discussed using subnetting to hide site LAN interconnection from networks outside the site. They introduce techniques such as ARP-based subnetting, ARP bridging, and extension of ARP itself. ARP bridging in particular is very similar in practice to the approach we take, although we assign a subnet per-host rather than per-LAN, and we manage address allocation and connectivity using protocols unavailable at the time.

VI. Conclusions

This paper has drawn upon previous user studies to reflect on the distinctive nature of home networks and the implications for domestic network infrastructures. Two particular user needs that arose from these studies were for richer visibility into and greater control over the home wireless network, as part of the everyday management of the home by inhabitants. We considered how to exploit the nature of the home network to shape how it is presented and opened to user control.

Simply put, the home is different to standard networking environments, and many of the presumptions made in such networks do not hold. Specifically, home networks are smaller in size, the equipment is physically accessible and access is often shared among inhabitants, and the policies involved are flexible and often dynamically negotiated. Exploiting this understanding allows us to move away from traditional views of network infrastructure, which must be tolerant of scale, physically distributed, and impose their policies on users.

We use the Open vSwitch and NOX platforms to provide flow-based management of the home network. As part of this flow-based management, we exploit the social conventions in the home to manage introduction of devices to the network, and their subsequent access to each other and Internet hosted services. This required modification of three standard protocols, DHCP, EAPOL and DNS, albeit in their behaviour only *not* their wire formats, due to the need to retain compatibility with legacy deployed stacks.

Our exploration suggests that, just as with other edge networks, existing presumptions could usefully be re-examined to see if they still apply in this context. *Do we wish to maintain net neutrality in the home?* Inhabitants do not appear to see network traffic as equal, often desiring imbalance in performance received by different forms of traffic. *Must the end-to-end argument apply?* Householders understand and exploit the physical nature of their home and use trust boundaries to manage access; we have exploited these resources to explicitly manage the network. *Should communication infrastructures remain separate from the devices that use them?* In the home setting this separation proves problematic as people, ranging from the home tinkerer to the DIY expert, wish to interact directly with the network as they do with other parts of their homes' physical infrastructures. Our exploration suggests use of a range of displays and devices existing not as clients exploiting the infrastructure but as extensions of the infrastructure making it more available and controllable.

Inability to understand and control network infrastructure has made it difficult for people to understand and live with it in their homes. We have developed a home router that both captures information about people's use of the network and provides a point of interaction to control the network. Our initial developments have explored the extent to which residents may be involved in some of the protocols controlling the network; other protocols suitable for modification are under consideration. We are currently involved in the deployment and study of use of our home router to better understand relevant user needs and how we might more systematically exploit the data we are collecting. We are also exploring how to use this data in other areas such as security and power management.

Acknowledgements

The research on which this paper is based was funded by the RCUK supported Horizon Hub, EP/G065802/1, and EPSRC Wired and Wireless Intelligent Networked Systems Initiative 'Homework,' EP/F064276/1.

References

[1] P. Tolmie, A. Crabtree, T. Rodden, C. Greenhalgh, and S. Benford, "Making the home network at home: digital housekeeping," in *Proceedings ECSCW*. Limerick, Ireland: Springer, Sep. 24–28 2007, pp. 331–350.

[2] T. Rodden, A. Crabtree, T. Hemmings, B. Koleva, J. Hunble, K.-P. Akesson, and P. Hansson, *Assembling connected cooperative residential domains*. Springer, 2007, pp. 120–142, in *The Disappearing Computer: Interaction Design, System Infrastructures and Applications for Smart Environments* (eds. Streitz, N., Kameas, A., and Mavrommati, I.).

[3] T. Rodden and A. Crabtree, "Domestic routines and design for the home," *Computer Supported Cooperative Work: The Journal of Collaborative Computing*, vol. 13, no. 2, pp. 191–220, 2004.

[4] T. Rodden, A. Crabtree, T. Hemmings, B. Koleva, J. Hunble, K.-P. Akesson, and P. Hansson, "Between the dazzle of a new building and its eventual corpse: assembling the ubiquitous home," in *Proc. ACM DIS*, Cambridge, MA, USA, 2004, pp. 71–80.

[5] A. Crabtree, T. Rodden, T. Hemmings, and S. Benford, "Finding a place for ubicomp in the home," in *Proc. UbiComp*. Seattle, WA, USA: Springer, Oct. 12–15 2003, pp. 208–226.

[6] E. Shehan and W. Edwards, "Home networking and HCI: What hath God wrought?" in *Proc. ACM CHI*, 2007.

[7] R. Grinter and W. Edwards, "The work to make the home network work," in *Proc. ECSCW*, Paris, France, Sep. 18–22 2005, pp. 469–488.

[8] J.-Y. Sung, L. Guo, R. Grinter, and H. Christensen, "My roomba is rambo: Intimate home appliances," in *Proc. UbiComp*, Innsbrück, Austria, Sep. 16–19 2007, pp. 145–162.

[9] M. Chetty, J.-Y. Sung, and R. Grinter, "How smart homes learn: The evolution of the networked home and household," in *Proc. UbiComp*, Innsbrück, Austria, Sep. 16–19 2007, pp. 127–144.

[10] E. Shehan-Poole, M. Chetty, W. Edwards, and R. Grinter, "Designing interactive home network maintenance tools," in *Proc. ACM DIS*, Cape Town, South Africa, 2008.

[11] P. Brundell, A. Crabtree, R. Mortier, T. Rodden, P. Tennent, and P. Tolmie, "The network from above and below," in *Proc. ACM SIGCOMM Workshop on Measurements Up the STack (W-MUST)*, Toronto, Canada, Aug. 2011, to appear.

[12] J. Sventek, A. Koliousis, O. Sharma, N. Dulay, D. Pediaditakis, M. Sloman, T. Rodden, T. Lodge, B. Bedwell, K. Glover, and R. Mortier, "An information plane architecture supporting home network management," in *Proc. Integrated Management (IM)*, 2011.

[13] R. Mortier, B. Bedwell, K. Glover, T. Lodge, T. Rodden, C. Rotsos, A. W. Moore, A. Koliousis, and J. Sventek, "Supporting novel home network management interfaces with OpenFlow and NOX," in *Proc. ACM SIGCOMM*, Toronto, ON, Canada, Aug. 15–19 2011, extended demo abstract.

[14] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in college networks," whitepaper, http://www.openflowswitch.org/wp/documents/.

[15] A. Arasu, S. Babu, and J. Widom, "The CQL continuous query language: semantic foundations and query execution," *The VLDB Journal*, vol. 15, no. 2, Jun. 2005.

[16] R. Droms, "Dynamic Host Configuration Protocol," IETF, RFC 2131, Mar. 1997.

[17] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowetz, and Ed., "Extensible Authentication Protocol (EAP)," IETF, RFC 3748, Jun. 2004.

[18] K. Calvert, W. Edwards, and R. Grinter, "Moving toward the middle: The case against the end-to-end argument in home networking," in *Proc. HOTNETS*, Atlanta, GA, USA, Nov. 14–15 2007.

[19] C. Swindells, K. Inkpen, J. Dill, and M. Tory, "That one there! pointing to establish device identity," in *Proc. UIST*, Paris, France, Oct. 27–30 2002, pp. 151–160.

[20] D. Balfanz, G. Durfee, R. Grinter, D. Smetters, and P. Stewart, "Network-in-a-box: how to set up a secure wireless network in under a minute," in *Proc. the 13th USENIX Security Symposium*, San Diego, CA, USA, 2004.

[21] J. Yang and W. Edwards, "Icebox: Toward easy-to-use home networking," in *Proc. INTERACT*, Rio de Janeiro, Brazil, Sep. 2007.

[22] J. Yang, W. Edwards, and D. Haslem, "Eden: Supporting home network management through interactive visual tools," in *Proc. UIST*, New York, NY, USA, Oct. 3–6 2010, pp. 109–118.

[23] D. Joumblatt, R. Teixeira, J. Chandrashekar, and N. Taft, "HostView: Annotating end-host performance measurements with user feedback," in *Proc. ACM HOTMETRICS*, New York, NY, USA, Jun. 18 2010.

[24] K. Calvert, K. Edwards, N. Feamster, R. Grinter, Y. Deng, and X. Zhou, "Instrumenting home networks," in *Proc. ACM HOMENETS*, New Delhi, India, Sep. 3 2010.

[25] E. Cooke, R. Mortier, A. Donnelly, P. Barham, and R. Isaacs, "Reclaiming network-wide visibility using ubiquitous endsystem monitors," in *Proc. USENIX Annual Technical Conference*, Boston, MA, USA, Jun. 2006, pp. 257–262.

[26] M. Casado, M. Freedman, J. Pettit, N. McKeown, and S. Shenker, "Ethane: Taking control of the enterprise," in *Proc. ACM SIGCOMM*, Kyoto, Japan, Aug. 27–31 2007.

[27] T. Karagiannis, R. Mortier, and A. Rowstron, "Network exception handlers: Host-network control in enterprise networks," in *Proc. ACM SIGCOMM*, Seattle, WA, USA, Aug. 17–22 2008.

[28] J. Mogul, "Internet subnets," IETF, RFC 917, Oct. 1984.

[29] J. Postel, "Multi-LAN address resolution," IETF, RFC 925, Oct. 1984.