An Information Plane Architecture Supporting Home Network Management

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

N. Dulay, D. Pediaditakis, M. Sloman
Department of Computing
Imperial College
London, UK
{n.dulay, d.pediaditakis, m.sloman}@imperial.ac.uk

T. Rodden, T. Lodge, B. Bedwell, K. Glover, R. Mortier
School of Computer Science
University of Nottingham
Nottingham, UK
{tar, txl, bzb, ktg, rmm}@cs.nott.ac.uk

Abstract—**Home networks have evolved to become small-scale versions of enterprise networks. The tools for visualizing and managing such networks are primitive and continue to require networked systems expertise on the part of the home user. As a result, non-expert home users must manually manage non-obvious aspects of the network - e.g., MAC address filtering, network masks, and firewall rules, using these primitive tools.**

The Homework information plane architecture uses stream database concepts to generate derived events from streams of raw events. This supports a variety of visualization and monitoring techniques, and also enables construction of a closed-loop, policy-based management system. This paper describes the information plane architecture and its associated policy-based management infrastructure. Exemplar visualization and closed-loop management applications enabled by the resulting system (tuned to the skills of non-expert home users) are discussed.

Keywords – component, information plane, home network, network management, network visualization, policy-based management

I. INTRODUCTION

Over 300 million people worldwide have home broadband connections to the Internet [1]. Many of these households are currently exploring the use of in-home wired and wireless networking, not only to allow multiple computers to share the broadband connection, but also to enable media sharing, gaming, and other new applications. However, despite the growing interest in home networking, these technologies remain extraordinarily difficult for people to install, manage, and use. Current approaches provide little support for end-user understanding and control of network technologies and the resulting difficulties are rapidly becoming a key roadblock to the deployment of next-generation applications in communication, healthcare, and entertainment [2].

Today's domestic infrastructures are opaque to users and prove clumsy and awkward in day-to-day use. This stems from the fact that the current suite of Internet protocols and architectures has migrated to the home with little or no reflection upon their appropriateness. The current home network is essentially built around the same protocols, architectures, and tools that were developed for the Internet as a whole in the 1970's [3]. Inherent in the Internet's 'end-to-end' architecture is the notion that the core is simple and stable, providing only a semantically neutral transport service. The Internet was designed for a certain context of use (assuming relatively trustworthy endpoints), made assumptions about its users (skilled network and systems administrators running the edge nodes and network core), and tried to accomplish a set of goals (e.g., scalability to millions of nodes) that are not directly appropriate for the home network.

The Homework project [4] is adopting a user-centred approach to the creation of the next generation domestic infrastructure that combines empirical understanding of use with a fundamental re-invention of the protocols, models and architectures of the domestic setting. Our research addresses several key objectives:

1. The development and assessment of a range of interactive techniques that make key features of the domestic infrastructure, including features associated with management, measurement and modelling, available to inhabitants in a way that reflects their needs and allows non-technical users to develop sufficiently rich understandings of the supporting system.
2. The investigation of new approaches to significantly reduce the overhead involved in the configuration and management of the infrastructure.
3. The development of new approaches to infrastructure measurement and monitoring that make key information about the supporting infrastructure available to users.

A key technical aspect of this next generation domestic infrastructure is an information plane architecture that flexibly enables real-time access to home network measurements of interest, performs composite event matching using these meas-

Research supported by the UK Engineering and Physical Sciences Research Council.

urements (as raw input events) to generate higher-level event notifications to drive management and/or visualization software, and persists the measurement data for offline analysis/use.

This paper describes the information plane architecture, its initial implementation and performance, and its associated policy-based management infrastructure. The efficacy of the resulting system is demonstrated through exemplar closed-loop management and visualization applications.

The contributions of this work are:

- A flexible, extensible information plane architecture has been designed for use in home network environments.
- This architecture has been instantiated in commodity PC hardware and software that replaces the domestic wireless router.
- This implementation has been integrated with a policy-based management engine to support autonomous management of home networks.
- The integrated system has been augmented with various components in preparation for deployment in users' homes to determine future human-centred changes to the system.

II. Related Work

A. Home Network Management

Network management is challenging due to the number of layers involved, the heterogeneity of technologies and the need to evolve networks over time. In home networks, these existing challenges are compounded by the need to provide tools for non-expert users to understand and manage their network. In [5] a modified home router was used to collect data that was used to drive an interactive visualization system to monitor and control bandwidth usage. HomeMaestro aimed at providing application fairness for a home network based on a set of application level weights elicited from users [6]. The necessity for traffic and bandwidth management was demonstrated in [7], while [8] showed how even simple changes to wireless settings can lead to significant improvements in performance.

There is substantial current interest in home network management, especially in the Human-Computer Interaction research community. A satellite workshop at SIGCOMM (Homenets) recently focussed on home networking issues bringing together a growing community of researchers addressing these key challenges. The Home Network Data Recorder [9] identifies many of the same issues for which the Homework information plane architecture has been devised.

B. Information Plane

As the networking community investigates the next generation Internet, it has become increasingly clear that a knowledge/information plane [10] that complements existing data and signalling planes is required. This third plane can be likened to a distributed database, where some of the content is raw data while other content is derived. For example, the Sophia system [11] provides an "information plane" for PlanetLab based upon a distributed system that collects, stores, propagates, aggregates, and reacts to observations about the network's current conditions. Through the provision of this distributed database, and sophisticated mechanisms for accessing the information, it is possible to provide a richer set of information about the workings of the network to support management, in general, and human-centred management, in particular.

C. Policy-based Management

In enterprise networks, policy-based management has gained acceptance as a flexible approach for addressing many requirements for configuration, security, performance and fault-tolerance [12]. Use in home networks has been limited to simple policies for router configuration, wireless channel selection [13], bandwidth management [14], and traffic prioritisation [15]. A policy-based management system that supports a wider range of policies and better models and abstractions for home networks and their users is needed.

D. Domestic Network Visualization

Network visualization has often been used to present information about the machines on a network and the overall topology [16]. The small scale of networks in domestic settings has meant the network visualization has allowed these to be used as the primary interface for managing the network. Network Magic [17] has built upon the lessons of network visualizations to present domestic networks topologically and to use this as the principle means of interfacing with the home network. Applications such as Eden [18] exploit the physical layout of the home as a means of showing the machines and their connections to each other.

III. The Information Plane

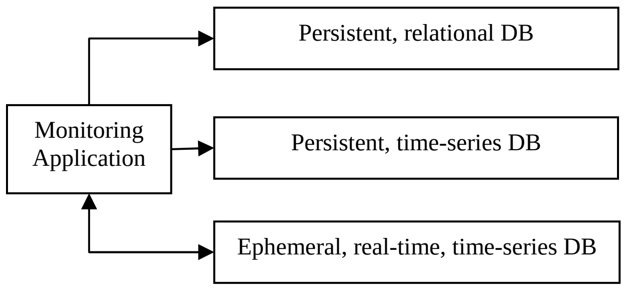
Persistent, relational DB
Monitoring Application
Persistent, time-series DB
Ephemeral, real-time, time-series DB

Figure 1. The three components of the Homework Information Plane

A. The Architecture

The information plane at the centre of the Homework monitoring and management system is an active, distributed database. The database is composed of three, distinct components, as shown in Figure 1.

The rationale for this structure is as follows:

- The ephemeral, real-time, time-series component is used to hold the most recently received measurements stored by different monitoring applications; the continuous, large volume of such measurements makes it impossible to persist the measurements, and requires optimization of resources to keep up with the volume of measurements; the primary ordering parameter of these measurements is the time of occurrence; additionally, this component must be able to detect specific patterns of behaviour as events arrive in order to notify relevant management components.

- The persistent, time-series component is used to hold time-series data that will be needed for subsequent analysis; specific monitoring components are used to obtain such data from the ephemeral database and store it appropriately; the data stored may be the raw data obtained from the ephemeral database, or it may be derived information generated from the raw data so obtained.
- Finally, some derived information may not be related at all to time of occurrence, in which case it is stored in a persistent, relational component.

In the diagram, an arrow (⇨) implies that the component at the tail of the arrow actively invokes the component at the head of the arrow. For example, the double-headed arrow (⬄) connecting the application and the ephemeral component indicates that the application actively invokes the ephemeral component; it also indicates that when a behaviour pattern to which an application has subscribed is detected, the ephemeral database actively invokes the application.

A system consists of a number of monitoring applications and instances of the information plane components. There are several roles that can be adopted by monitoring applications:

- Population – applications performing this role measure some aspect of the system and insert the resulting measurement data into the ephemeral component.
- Persistence – applications performing this role extract data from the ephemeral database in order to store that data, or information derived from that data, into the persistent time-series component.
- Reaction – applications performing this role register interest in particular behaviour patterns; when such a pattern is detected by the ephemeral component, it notifies the application of the occurrence to enable it to react to the event.
- Display – applications performing this role extract real-time data from the ephemeral database for display to a user of the system.

B. *The Design*

It should be apparent from the discussion above that the ephemeral component is at the heart of this architecture. A stream database [19], with integrated complex event stream processing, is the implementation technology chosen to meet the needs of the various application roles. This section provides additional detail on the design of the ephemeral component.

The scope of queries for most stream databases is the set of tuples that pass an observation point during a window in time. In our system, the ephemeral component has a fixed-size memory buffer into which tuples are stored as they arrive, with appropriate control structures used to link these tuples into tables. This memory buffer is treated in a circular fashion; when the write pointer for the buffer overtakes the first used location, the oldest tuple[s] are removed from the buffer to make room for new tuples. The memory buffer is appropriately-sized to enable retention of tuples for a minimum period of time before they need to be overwritten.[1]

The architecture, with its dependence upon many monitoring applications interacting with the database components, requires a lightweight and dependable interaction mechanism between application and information plane components.

C. *The Implementation*

Given the goals of the Homework project described in Section I, we have focused on an implementation of the architecture in a prototype, Linux-based Homework wireless router for deployment in users' homes. The Homework router is an Atom 1.6GHz EeePC 1000H netbook with 2GB of RAM running Ubuntu 10.04. The ephemeral database component runs as a process in the router, as do population, persistence, and reaction applications. Display applications can be run on any device that is connected to the router, either directly over the wireless link, or through the backhaul network if the router's firewall rules enable such interaction.[2]

Interaction between applications and the ephemeral component is facilitated by a lightweight, UDP-based remote procedure call implementation for which we have generated C/C++ and Java bindings that have been tested on Linux, Windows/Cygwin, OS/X, and Android systems. The RPC system facilitates the exchange of request packets for response packets, supports one-outstanding-packet semantics for each binding, supports fragmentation/reassembly of large buffers, optimizes ACKs for rapid request/response exchanges, and maintains liveness for long-running exchanges. Performance of the RPC system is typical: between processes on the router, it is limited by the thread-scheduling context switch time; between a process on a handheld and another on the router, it is limited by transmission speed and error characteristics of the communication medium. Construction using unreliable datagrams is consistent with a link-level implementation, if performance and resource constraints demand it.

The ephemeral component implements an extended version of the Stanford STREAM Data Manager [19], which includes:

- Retention of events based upon the capacity of the circular memory buffer.

[1] One goal of the project is to enable homeowners to contract with 3rd party support organizations to fix home networking problems. We currently anticipate that having access to all events during the past 2 hours (subject to validation using industrial partner log data) will be sufficient to enable this support. To guarantee that the support organization has access to that data when a customer makes such a trouble call, the ephemeral component must support a snapshot capability that is triggered by the user when a problem is detected and before the trouble call is placed.

[2] Population, persistence, and reaction applications can also execute on machines connected to the router if firewall rules permit.

- Extensions to the Continuous Query Language (CQL) [20] to support a richer set of time window expressions[3].
- Ability for an application to register interest in the occurrence of particular patterns of events and to receive notifications when a pattern is detected.

The fast snapshot requirement (see footnote 1 above) is implemented exploiting the copy-on-write semantics of Linux fork(); upon receipt of a request to create a snapshot, the ephemeral component forks a copy of itself; the parent waits for the child copy to be operational, then continues to accept requests from its monitoring applications; the child copy advertises its service on a different UDP port, and it refuses to execute any CQL commands that would modify the in-memory database. On the Homework router, the snapshot version of the database is created and the parent resumes operation within 10ms for every 100MB of circular buffer (e.g. for a 1.5GB buffer, the dead time is 150ms).

The persistent, time-series component is implemented using the BerkeleyDB system [21]. For each table that is maintained in the ephemeral database, a custom persistence application extracts all tuples at periodic intervals, and then writes these to a BerkeleyDB file. A query processor for the extended CQL has been implemented that operates upon the persistent data in the Berkeley DB files.

TABLE I. THE HOMEWORK DATABASE SCHEMAS.

Table	Attributes	Description
Flows	Protocol, src IP addr, src port, dst IP addr, dst port, # packets, # bytes	A tuple contains the number of packets, and the number of bytes associated with a particular flow in the last second.
Links	MAC address, RSSI, # retries, # packets	A tuple contains the average received signal strength, the number of retries, and the number of packets associated with a particular host in the last second,
Leases	Action, MAC address, IP address, host name	A tuple denotes either that a DHCP lease has been granted to a particular host (action is "add"), or that a lease has been revoked (action is "del").

IV. THE POLICY-BASED MANAGEMENT ARCHITECTURE

A. *The Policy Engine*

The Homework policy engine is a multi-threaded interpreter that runs as a background daemon alongside the databases in the information plane. It accepts policies from other Homework components, in particular, UI applications, and translates them into one or more of the following:

- Networking commands that, for example, call a firewall or traffic shaping service. Translation is carried out twice. The first parses and checks the policy for legality and consistency. The second is carried out (just-in-time) when a policy is enabled, verifying and validating the policy and converting it into system-specific commands for enforcement.
- Management objects within the policy engine that subscribe and respond to events from the ephemeral database. Management objects are particularly useful for more complex management tasks where the resources that need to be monitored are related to the overall network state (e.g. topology), the state of queues at the gateway, the quality of network links, the level of utilization of several network resources, as well as time of day, etc. Management objects typically respond to events by invoking or revoking networking commands, inserting data into the ephemeral or persistent databases, enabling and disabling policies. In addition, management objects can also process events in a goal-driven manner using Nilsson's teleo-reactive semantics [22].

B. *Policies*

Two classes of policy are currently supported:

- Access control policies that model who (subjects) can access which resources (services) under what conditions. Hierarchical containers called domains are used for grouping subjects, roles, resources, services etc. Access control policies map to firewall commands unless a condition is given that isn't supported by the firewall, in which case a special condition object needs to be compiled for the firewall. For some conditions, for example, an access control policy that permits access to a website only at certain times of the day, a standard management object is used that dynamically adds or removes the firewall rule at the beginning and end of the time period.
- Traffic control policies are used to shape or prioritise traffic. More specifically, they allow us to (i) define a hard upper limit (bandwidth cap) on the upload and/or download data rates, (ii) set a minimum assured data rate and (iii) assign a priority to the traffic. Unsurprisingly, the network demands of users and applications are outstripping the provisioning of network providers. Traffic control policies are needed to allow users to both maximise and share their available bandwidth, and also to ensure good performance for low-latency applications, particularly in periods of over-utilisation.

C. *The Implementation*

The Homework policy engine is a development of the Ponder2 policy engine that has been successfully used in a variety of distributed and ubiquitous settings [23]. It's implemented in Java and its bytecode is less than 1MB. Homework policies are written in JSON and grouped into policy sets. The databases in the information plane are used to store policy sets and log policy events such as loading and enabling of policies. When the policy engine is started it initialises itself from the most recent enabled policy sets in the persistent time-series database.

Access control policies are mapped to custom chains in Linux iptables and 'jump'ed to as needed. Traffic control rules are mapped to a combination of iptables and tc commands. A packet marking scheme is employed (using the mangle table and prerouting chain) with traffic redirected to the appropriate traffic classes using the HTB classful queuing discipline and tc filters. The choice to use iptables and tc was made because

[3] "SINCE <timestamp>" and "INTERVAL LB <timestamp>,<timestamp> RB", where LB is '(' or '[' and RB is ')' or ']'

they are robust and offer a wide range of configuration options. It is relatively easy however to write new backends for alternative networking technologies, like OpenFlow [24] and NOX [25].

Note that the policy engine is not only responsible for invoking the commands for a particular policy but it can also revoke them, for example deleting firewall rules, stopping netfilter marking and deleting tc qdiscs/classes/filters.

V. Exemplar Applications

We have developed a number of applications motivated by users concerns that arose from a series of ethnographic studies of home networks in domestic settings [26]. These studies highlighted the opaque nature of existing domestic network technology as particularly problematic for users. Current routers provide little information about the nature of the network to inhabitants [3] with the result that they are uncertain about the current status of the network, the nature of any particular issues or how they might address these. Furthermore, control interfaces require significant technical knowledge and expertise. Our developed applications make key features of the network infrastructure available to users in a manner that is understandable by inhabitants allowing them to make judgments about the nature of the network and provide simple interfaces to exercise direct control of the network. The challenge is how we map from the technical information provided by the information plane to representations that are easy to understand with limited knowledge of networking technologies. All of the examples discussed below are with respect to the database schemas shown in Table I.

A. Elephant flow detection and control

Network flows have been classified as either “elephants” or “mice” [27]. Elephants are the few flows that not only contribute to the overall traffic volume (both in terms of number of packets and number of bytes), but also show significant persistence in time [28]. Mice, on the other hand, are the many short-lived, small flows in the network.

An exemplar closed-loop management application has been developed to exploit the active nature of the ephemeral, real-time, time-series component of the Homework information plane. The ***FlowManager*** monitors and manages flows in real-time. Managed flows are the “elephants” in the home, i.e. those flows that deprive other, perhaps smaller, flows of available bandwidth during a given period of time.

The ***FlowManager*** is a reaction application (cf. Section III.A) and has resulted from the integration of the Homework monitoring system and the Homework policy engine. ***FlowManager*** is a policy management object which registers an interest with the database for those flows (5-tuple identifiers) which are active during a time window W and whose rate exceeds a threshold T.

The Homework database maintains a rate (an exponential moving average in bytes/sec) for each active flow in the home network, which is updated every five seconds. The ***FlowManager*** expresses an interest by selecting active flows which satisfy a specific predicate which can be expressed both in terms of attributes (i.e. protocol type, source IP address, source port, destination IP address, destination port, current rate) and in terms of a time window. Whenever an active flow matches the specified predicate, the Homework database sends a new event (adds a new tuple) to the ***FlowManager***. Upon receipt of such an event, the ***FlowManager*** generates and submits an appropriate traffic control policy document for processing by the policy engine (see Annex 1 for an example of such a policy). For flows that no longer match any predicate, a corresponding delete event is sent; upon receipt of a delete event, the corresponding policy is disabled. The following experiment, using iperf to simulate flows in the home, was performed to validate the efficacy of the aforementioned management system.

Suppose that the “elephants” in the home network are TCP iperf flows, and the ***FlowManager*** expresses an interest for those iperf flows whose rate exceed 1Mbps, and which are active during a time window of 300 seconds in the future; the default management action is to allocate no more than 1Mbps to each of the managed flows.

Six iperf flows start and stop before, during, and after the time window as shown in Figure 2. Six iperf (version 2.0.4) clients reside within the 192.168.9.0/24 subnetwork, our prototype home network. They connect to the homework router via an 802.11g wireless link. Six iperf servers (one for each flow) reside on a host outside the home network. The link from the router to the iperf servers is 100-baseT Ethernet. Both the Homework database and the FlowManager run on the Homework router.

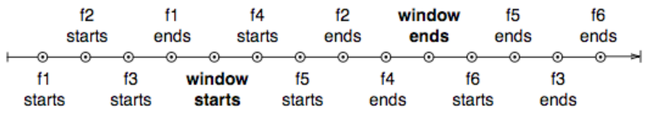

Figure 2. The duration of the experiment is thirteen minutes. Events (circles in the diagram) are spaced one minute apart.

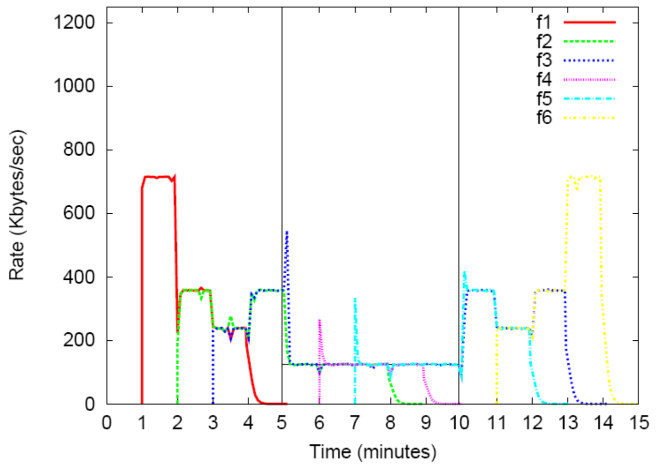

Figure 3. Those iperf flows that are active during the 300-second time window (vertical lines in the graph) are rate-limited to 1Mbps

Figure 3 illustrates the results. Before the time window starts, flows f1, f2, and f3 share the available bandwidth without restriction. During the time window (vertical lines in the graph), flows f2, f3, f4, and f5 are rate-limited to 1Mbps; the remainder of the bandwidth is available to any unmanaged flows for sharing. When the window ends, the list of managed flows is empty (the FlowManager has received delete events for each of f2, f3, f4, and f5). After the window ends, flows f4 and f5 (later joined by f6) resume their normal operation.

In summary, the ***FlowManager*** application is a reaction application that enables users to actively manage their bandwidth. For example, they can now express policies of the form "limit Junior's file downloads while Dad is working from home, between 9am and 5pm." In order to explore how we present this information to household inhabitants, we have developed a range of display applications and interfaces.

B. The network control interface

This user-focused application explores how we might provide users with a central point of awareness and control for the network. This exploits people's everyday understanding of control panels in their home (e.g. heating or alarm panels). Our approach builds upon existing work such as the ICE panel [29] that have used the physical arrangement of a central control panel to manage network introduction and removal. Our interface runs on a dedicated touch screen in the home and we exploit this physical arrangement to provide a focal point for people to view current network status and manage the network.

Our emphasis is on providing a readily available overview of the machines on the network and conveying to users the current status of these machines. We also wish to provide lightweight management facilities to allow householders to manage the devices on the network. For example, we wish to allow inhabitants to restrict access to the network or to remove machines from the network. Given the heterogeneity of the machines on home networks, we are particularly interested in providing information on the status of machines on the network and in enabling management and control without installing any additional software on these machines.

The basic version of this interface provides users with a real time display of the current status of the network (Figure 4). Devices connected to the network are shown on the right of the screen while devices that are within range of the wireless network but are not connected are shown on the left. This display shows four devices connected to the network and a single device that is within range but is not currently connected.

As devices come within range of the home router they appear in the appropriate area of the screen and are updated in real time. A machine will initially appear on the left of the display and as a user connects to the network it moves to the right of the display. The display also dynamically maps key network characteristics of the devices to particular display features of the labels representing the machines providing an at-a-glance animation of the network. Key mappings in the current display are:

- Wireless signal strength is mapped to the transparency level of the machine label giving the effect of machines with a weak signal fading in the background
- The proportion of bandwidth usage for each machine is mapped to label sizes. Thus, in Figure 4, the Squeezebox Receiver is using the largest proportion of the bandwidth. This mapping is animated in real-time with machine labels growing/shrinking in size.
- Packet retries associated with each machine are displayed as red highlights on the label showing those machines that are experiencing traffic loss issues.

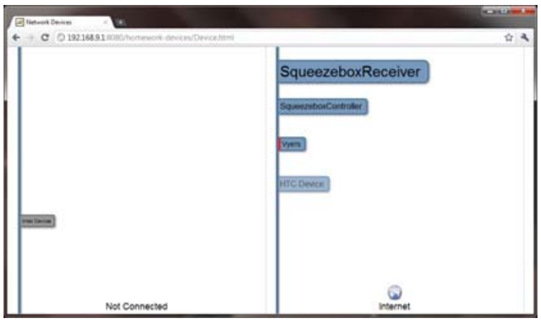

Figure 4. The basic control panel

Selecting a machine label on the display causes a popup label to appear providing more details of the machine and the nature of the network information. For example, manufacturer details and current data transfer information for the machine *Vyers* is shown in Figure 5.

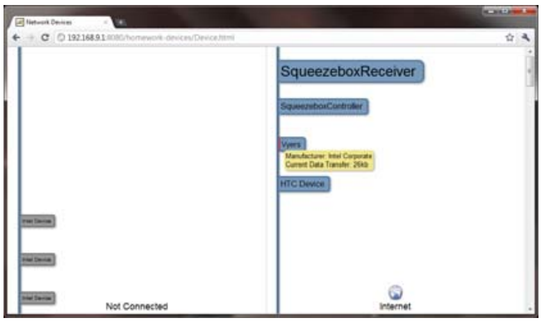

Figure 5. More details for a particular machine

A more elaborate version of the interface provides simple control capabilities where users can assert control over the traffic flowing through the home network using a simple drag and drop approach. In this version of the interface, users can bar particular machines from the network by dragging their icon from the right hand side of the machine to the unconnected area on the left. The interface also allows control of traffic on machines through a simple control interface, which allows users to affect particular equivalence classes of application protocols on a device. The control pane for the machine *Vyers* is shown in Figure 6, allowing users to selectively turn off or limit a number of these equivalence classes.

This application was developed in HTML 5 allowing it to run across a number of displays. The control actions enabled by the application are communicated to the policy engine through the ephemeral component; this enables the request to be logged persistently, as well as enabling the request to be pushed to the policy engine.

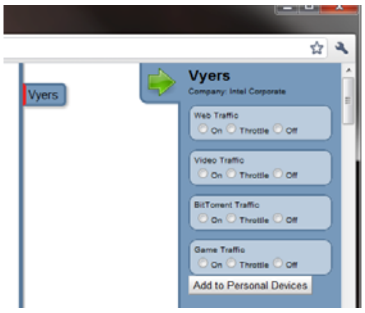

Figure 6. The control pane for a machine on the network

C. *Bandwidth Contention*

Another important display application focuses on understanding issues of bandwidth contention in the home network. The impact on household members of others running particularly bandwidth-intensive applications was a common phenomenon that we observed in a number of households. Understanding "who is hogging the bandwidth" was important to users to allow them to negotiate access to the network. Similar to the approach followed by others [5], we wish to provide a simple display that allows household residents to understand which machines and applications are consuming bandwidth. This can enable the inhabitants to negotiate between themselves about access to the network to resolve any particular contention, without automated bandwidth management.

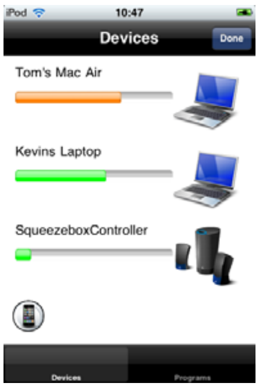

Figure 7. Bandwidth consumption per machine

This application is designed to run on a portable device allowing individuals to explore the use of the network and the characteristics of the display. The interface highlights the top three machines in terms of bandwidth consumption. As this proportion increases beyond 50%, the line indicating proportion of bandwidth used progressively changes colour from green to red, as shown in Figure 7, where the machine 'Tom's Mac Air' is consuming the highest proportion of bandwidth.

Selecting a particular machine from this interface allows users to drill down to gain further information. Figure 8 shows an ordered list of the particular protocols associated with that machine displaying the proportion of the bandwidth being used by each protocol. This data is drawn from the information plane and updated in real-time, allowing users to view the impact of their actions on the network as they change their behaviours by pausing applications.

This application is developed to run on smart phones (in this case iPhones) and, like the other interfaces and applications, takes its information from the ephemeral component.

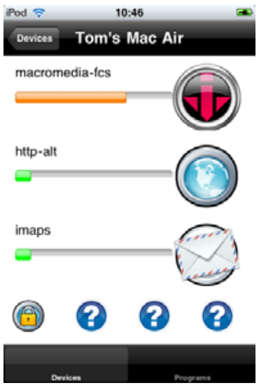

Figure 8. Bandwidth usage per protocol for 'Tom's Mac Air'

VI. Discussion

The prototype has been deployed in the homes of several project members, as well as experiencing production use in the Glasgow School of Computing Science. Besides functioning as an effective wireless router, the system has enabled experimentation with different types of population, persistence, reaction and display applications (those described above are just an exemplary subset). The system is now being fine-tuned and ruggedized for deployment into homes of non-network experts.

The ephemeral component of the information plane exhibits the flexibility required to support known and future requirements for our research. The decision to divorce persistence of tuples from the responsibilities of the ephemeral component has enabled us to focus on maximizing the performance of that component; it also enables us to embed the component in the Linux kernel in the future. (Since many of the events that populate the tables originate from within the kernel, this embedding will substantially reduce the current overheads induced by using libpcap to obtain information about individual packets.)

The CQL time window extensions have proved useful in facilitating pull-style retrieval of tuples for persistence and display applications. The ability to register interest in changes to a table (i.e. a select query is executed whenever the table tuple set changes and non-zero results are sent to the subscribing application) has enabled a form of pattern subscription that we have used to implement a number of reaction applications.

VII. Conclusions and Future Work

This paper has presented an Information Plane architecture for home networking. The architecture has been realised using commodity machines and software, enabling us to replace the home router with an equivalent system running the information plane. This arrangement has allowed us to provide significant information about the nature of the network to users. We also exploit a policy engine to allow control of the network through this modified router. A number of user-centred interfaces have been developed to provide users with network information and to allow them to control network traffic.

The current pattern subscription mechanism is too primitive for general use. We are evaluating several, different complex event processing systems with respect to meeting our active database requirements [30, 31]; the best match will be implemented in the ephemeral component.

We are currently in the process of deploying this information plane across a number of households to allow us to capture a range of data about traffic on these networks. This investigation will also explore the effectiveness of our user interfaces in order to allow us to refine them to ensure they are understood and used in practice by household members. Finally, we are exploring how we might make the data from our information plane remotely available to experts providing advice and help.

ANNEX 1 – EXAMPLE TRAFFIC CONTROL POLICY

```
{"RuleSet": [ {
        "Action":{
                "TrafficControl":{
                        "TrafficAction":"MaximumRate",
                        "Direction":"Upload",
                        "DataRate":1000,
                        "DataRateUnit":"Kbps"
                }
        },
        "Subject":[ {
                "AddressType":"IPv4",
                "Address":"192.168.9.55"
        } ],
        "Service":{
                "EndpointAddresses":[ {
                        "AddressType":"IPv4",
                        "Address":"130.209.253.156"
                } ],
                "L4Protocols":["tcp"],
                "SourcePorts":[42347],
                "DestinationPorts":[5001]
        }
} ] }
```

REFERENCES

[1] http://www.internetworldstats.com/dsl.htm

[2] The Consumer Electronics Association (CEA) (http://www.ce.org) report that home networking equipment is the most returned consumer electronics item with return rates for new products in excess of 25%.

[3] Yang, J., Edwards, W.K., “A Study on Network Management Tools of Householders”, Proc. Homenets, New Delhi, India, September 2010.

[4] http://www.homenetworks.ac.uk/

[5] Chetty, M., Banks, R., Harper, R., Regan, T., Sellen, A., Gkantsidis, C., Karagiannis, T., Key, P., “Who's Hogging the Bandwidth?: The Consequences of Revealing the Invisible in the Home”, Proc. CHI 2010, Atlanta, GA, USA, April 2010.

[6] Karagiannis T., et al., “HomeMaestro: Order from chaos in home networks”, Microsoft Research Report MSR-TR-2008-84, May 2008.

[7] Harrop W., Armitage G., “Quantifying the broadband access bandwidth demands of typical home users”, Proc. Australian Telecom. Networks & Applications Conference, Australia, December 2006.

[8] Papagiannaki K., Yarvis M., Conner W., “Experimental characterization of home wireless networks and design implications”. Proc. IEEE Infocom, Barcelona, Spain, April 2006.

[9] Calvert, K.L., Edwards, W.K., Feamster, N., Grinter, R.E., Deng, Y., Zhou, X., “Instrumenting Home Networks”, Proc. Homenets, New Delhi, India, September 2010.

[10] Clark, D., Partridge, C., Ramming, J., Wroclawski, J.,” A Knowledge Plane for the Internet”, Proc. SIGCOMM, Karlsruhe, Germany, August 2003.

[11] Wawrzoniak, M., Peterson, L., Roscoe, T., “Sophia: an Information Plane for networked systems”, ACM SIGCOMM Computer Communication Review, 34(1), pp. 15-20, January 2004.

[12] Bandara A., Damianou N., Lupu E., Sloman M. and Dulay N., “Policy-Based Management”, Handbook of Network and System Administration (eds. Bergstra J. and Burgess M.), Elsevier, Nov 2007.

[13] Pediaditakis D., Mostarda L., Dong C., Dulay N., “Policies for Self Tuning Home Networks”, 10th IEEE Intl. Symposium on Policies for Distributed Systems and Networks, July 2009.

[14] Jha S. and M. Hassan N. “Java implementation of policy-based bandwidth management”, Int. J. Network Management, Vol 13, No 4, Pages 249-258, 2003.

[15] Ibrahim, A., Foghlú, M., “Policy-based network management in home area networks : Interim test results”, Proc. 3rd Intl. Conf. on New Technologies, Mobility and Security (NTMS 2009), December 2009

[16] Marty, R., "Applied Security Vizualiation", Addison-Wesley, 2009.

[17] See www.purenetworks.com

[18] Yang,J., Edwards, W.K., Haslem, D., “Eden: Supporting Home Network Management Through Interactive Visual Tools”, Proc. of the 23rd ACM Symposium on User Interface Software and Technology, New York, NY, USA, October 2010.

[19] Babcock, B., Babu, S., Datar, M., Motwani, R., and Widom, J., “Models and issues in data stream systems”, Proc. of the Symposium on Principles of Database Systems, Madison, Wisconsin, USA, June 2002.

[20] Arasu, A., Babu, S., Widom, J., “The CQL continuous query language: semantic foundations and query execution”, International Journal of Very Large Data Bases (The VLDB Journal), 15(2), June 2005.

[21] Olson, M., Bostic, K., and Seltzer, M., “Berkeley DB”, Proceedings of the FREENIX Track: USENIX Annual Technical Conference, Monterey, CA, USA, June 1999.

[22] Nilsson N., “Teleo-Reactive Programs and the Triple-Tower Architecture”, Electronic Transactions on Artificial Intelligence, Vol. 5 (2001), Section B, pp. 99-110.

[23] Twidle K, Dulay N, Lupu E, Sloman M., “Ponder2: A Policy System for Autononmous Pervasive Environments”, 5th Intl Conference on Autonomic and Autonomous Systems, April 2009.

[24] McKeown, N., Anderson, T., Balakrishnan, H, Parulkar, G., Peterson, L., Rexford, J., Shenker, S., Turner, J., “OpenFlow: Enabling Innovation in Campus Networks”, http://www.openflowswitch.org/documents/openflow-wp-latest.pdf, accessed 12 September 2010.

[25] Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N., Shenker, S., “NOX: towards an operating system for networks”, SIGCOMM Comput. Commun. Rev. 38(3), July 2008.

[26] Grinter, R., Edwards, K., Chetty, M., Poole, E., Sung, J.Y., Yang, J. & Crabtree, A., Tolmie, P., Rodden, T., Greenhalgh, C., Benford, S., “The ins and outs of home networking: the case for useful and usable domestic networking”, ACM Trans. Comput.-Hum. Interact., 16 (2) 1-28, June 2009.

[27] Guo, L., Matta, I., “The war between mice and elephants”, Proc. Of the 9th International Conference on Network Protocols, pp. 180-188, November 2001.

[28] Papagiannaki, K., Taft, N., Bhattacharyya, S., Thiran, P., Salamatian, K., Diot, C., “A pragmatic definition of elephants in internet backbone traffic”, Proc. of 2nd ACM SIGCOMM Workshop on Internet Measurement, pp. 175-176, 2002.

[29] Yang, J., Edwards, W.K., “ICEbox: Toward Easy-to-Use Home Networking”, Proc. of Interact07, Rio de Janeiro, Brazil. September, 2007.

[30] Golab, L., Tamer Özsu, M., “Update-pattern-aware modelling and processing of continuous queries”, Proc. of ACM SIGMOD intl. conf. on Management of data, Baltimore, Maryland, USA, pp. 658-669, 2005.

[31] Brenna, L., Demers, A., Gehrke, J., Hong, M., Ossher, J., Panda, B., Riedewald, M., Thatte, M., White, W., “Cayuga: a high-performance event processing engine”, Proc. of ACM SIGMOD intl. conf. on Management of data, Beijing, China, pp. 1100-1102, 2007.