Software-defined wireless mesh networks for internet access sharing

Ahmed Abujoda\textsuperscript{a,}\textsuperscript{*}, David Dietrich\textsuperscript{a}, Panagiotis Papadimitriou\textsuperscript{a},
Arjuna Sathiaseelan\textsuperscript{b}

\textsuperscript{a} Institute of Communications Technology, Leibniz Universität Hannover, Germany
\textsuperscript{b} Computer Laboratory, University of Cambridge, UK

\textbf{Abstract}

Universal access to Internet is crucial, and as such, there have been several initiatives to enable wider access to the Internet. Public Access WiFi Service (PAWS) is one such initiative that takes advantage of the available unused capacity in home broadband connections and allows Less-than-Best Effort (LBE) access to these resources, as exemplified by Lowest Cost Denominator Networking (LCDNet). PAWS has been recently deployed in a deprived community in Nottingham, and, as any crowd-shared network, it faces limited coverage, since there is a single point of Internet access per guest whose availability depends on user sharing policies.

To mitigate this problem and extend the coverage, we use a crowd-shared wireless mesh network (WMN), at which the home routers are interconnected as a mesh. Such a WMN provides multiple points of Internet access and can enable resource pooling across all available paths to the Internet backhaul. In order to coordinate traffic redirections through the WMN, we implement and deploy a software-defined WMN (SDWMN) control plane in one of the CONFINE community networks. We further investigate the potential benefits of a crowd-shared WMN for public Internet access by performing a comparative study between a WMN and PAWS. Our experimental results show that a crowd-shared WMN can provide much higher utilization of the shared bandwidth and can accommodate a substantially larger volume of guest traffic.

\textcopyright 2015 Elsevier B.V. All rights reserved.

\section{Introduction}

The Internet has evolved into a critical infrastructure for education, employment, e-governance, remote health care, digital economy, and social media. However, the Internet today is facing the challenge of a growing digital divide, i.e., an increasing disparity between those with and without Internet access [60]. Access problems often stem from sparsely spread populations living in physically remote locations, since it is simply not cost-effective for Internet Service Providers (ISPs) to deploy the required infrastructure for broadband Internet access in these areas. Coupled with physical limitations of terrestrial infrastructures (mainly due to distance) to provide last mile access, remote communities also incur higher costs for connection between the exchange and backbone network when using wired technologies, because the distances are longer. Ubiquitous mobile broadband coverage is currently not feasible, since direct investment in local infrastructure is uneconomic [55]. Addressing digital exclusion due to socio-economic barriers is also important. The United Nations revealed the global disparity in fixed broadband access, showing that access to fixed broadband in some countries costs almost 40–100 times their national average income [34].

The reluctance of network operators (who are economically motivated) to provide wired and cellular infrastructures
to rural/remote areas have led to several initiatives to build large-scale, self-organized, and decentralized community wireless networks that use WiFi mesh technology (including long distance), due to the reduced cost of using the unlicensed spectrum [51]. These community wireless mesh networks have self-sustainable business models, which provide more localized communication services, as well as Internet backhaul support via peering agreements with traditional network operators who see such networks as a way to extend their reach at a lower cost. There are also community-led wireless initiatives such as crowd-shared wireless networks, in which home broadband owners share a portion of their home broadband with friends, neighbors, or other users either for free or as part of a service offering by the ISP (e.g., [6,53]).

Public Access WiFi Service (PAWS) [53] is a community-led crowd-shared WiFi service that uses a set of techniques that make use of the available unused capacity in home broadband networks and allowing Less-than-Best Effort (LBE) access to these resources, based on the Lowest Cost Denominator Networking (LCDNet) paradigm [58]. PAWS adopts an approach of community-wide participation, where home broadband subscribers are enabled to donate controlled but free use of their high-speed broadband Internet to fellow citizens. PAWS was deployed with 20 custom-made PAWS routers placed in a deprived community in Nottingham and was also trialed out in rural Wales. PAWS is essentially a crowd-shared access network (similar to FON [5]) for under-privileged users in urban and rural communities. PAWS has been facing ongoing deployment challenges, such as limited coverage, stemming from user sharing patterns. In particular, during the PAWS trial deployment, it was observed that home users did not share their broadband connection over periods at which either the whole bandwidth or all the ports of the home router were needed (i.e., PAWS uses an access point connected to the home router for Internet access sharing). Essentially, PAWS is a crowd-shared network with a single point of access per guest, and as such, Internet access sharing is highly dependent on user sharing policies (i.e., the periods at which user share their Internet connection).

To mitigate this problem, we investigate the potential benefits of extending PAWS or any crowd-shared wireless network to a wireless mesh network (WMN) by interconnecting wireless home routers. As such, a crowd-shared WMN provides extended coverage via multiple points of access for each guest. We particularly consider crowd-shared WMNs in residential areas, taking advantage of the dense deployment of wireless home routers. The main challenge in the management of such a WMN lies in the coordination of guest traffic redirections, such that the shared bandwidth is efficiently utilized. More precisely, traffic redirection requires the assignment of guest flows to gateways and the selection of paths (through the WMN) that provide sufficient capacity and low delay. Furthermore, decisions for traffic redirections should be also based on user sharing policies, when these are disclosed in advance. Given the amount of information that has to be collected before flow assignments can be made, we deem a centralized control plane as a more suitable approach to WMN management for Internet access sharing, since all information can be conveyed to a centralized controller facilitating the coordination of traffic redirections. In this respect, we leverage on software-defined networking (SDN) principles for the control plane design.

Along these lines, our contributions are the following:

- We present the design and implementation of a SDN control plane for the coordination of guest traffic redirections through the WMN.
- We develop an algorithm for the assignment of gateways to flows that require redirection, taking into account the flow demands, the residual bandwidth in the access links and the WMN, as well as, the advance knowledge of sharing policies.
- We generate a model for user sharing patterns based on the router on/off periods captured from the PAWS deployment in Nottingham.
- We quantify the benefits of a crowd-shared WMN for Internet access sharing using a deployment of our SDN control plane in Athens Wireless Metropolitan Network (AWMN) [4], i.e., one of the community networks of the CONFINE project [2,3,24]. Community Lab, https://community-lab.net/ In this respect, we perform a comparative study between a WMN and a crowd-shared network with limited coverage (i.e., one access point per guest).

This paper extends our previous study on software-defined WMNs, at which the efficiency of a crowd-shared WMN was assessed using simulations [18]. The remainder of the paper is organized as follows. In Section 2, we provide an overview of the software-defined WMN (SDWMN) control plane. Section 3 elaborates on techniques for guest traffic redirection. In Section 4, we discuss the implementation of the SDWMN and its deployment in AWMN. In Section 5, we present our evaluation results and discuss the benefits of a crowd-shared WMN for Internet access sharing. Section 6 discusses related work. Finally, Section 7 highlights our conclusions.

2. Software defined crowd-shared wireless mesh networks

The underlying problem with PAWS or any crowd-shared network is that they serve as single point of Internet access to guests within the coverage of the wireless router and hence, they have no provision to extend the coverage when no bandwidth is being shared. Based on our experience from the trial PAWS deployment, PAWS routers were not available for certain periods, because sharers needed all the bandwidth of their broadband connection or due to other reasons, such as economic constraints placed on home users in underprivileged areas where they are enforced to conserve energy by turning off the routers at nights. These observed user behaviors entail significant challenges for the successful adoption of PAWS.

A potential solution to this problem is to extend the PAWS network as a crowd-shared WMN. Such a network would allow home network users to share part of their own broadband connection with the public for free while also connected to each other as a WMN providing extended coverage (Fig. 1). Extending PAWS to a crowd-shared WMN departs from the norm: multiple users from different ISPs form part
of the WMN to provide free Internet connectivity, while most wireless community WMNs today are operated by a single organization. This raises important questions regarding the operation, configuration, and management of crowd-shared WMNs.

SDN can facilitate the management and operation of wireless networks at large scale. Leveraging on SDN’s centralized control and network-wide visibility, the management and operation of a crowd-shared WMN can be outsourced to a third party. In [54], we describe a holistic approach of coupling both social and economic incentives in designing future networks allowing the extension of the stakeholder value chain to include more than the two traditional parties (consumer and Internet Service Provider). Compared to existing mesh networks for Internet access sharing (e.g., Freifunk [6]), our approach provides more opportunities for non-governmental organizations and local governments (driven by social goals rather than economic) to become virtual network operators. Enabling a third party to federate such wireless home networks would reduce the operating expenditures for network operators as well as enable new economic models for revenue generation from currently underutilized infrastructures.

In particular, we rely on SDN to create the notion of Virtual Public Networks (VPuN), i.e., crowd-shared home networks created, deployed and managed through an evolutionary SDN control abstraction [54]. Although originally intended for crowd-shared wireless networks such as PAWS, VPuN can be also used for crowd-shared WMNs, enabling resource pooling across multiple home broadband connections based on the prevailing network conditions and usage sharing patterns. Based on the notion of VPuN, we have deployed a SDWMN control plane in AWMN [4]. As part of the Community-Lab [3], AWMN consists of more than 1000 wireless nodes and 30 research devices (RDs) that host isolated containers (i.e., so-called slivers) which can be allocated and controlled by a user, according to the needs of his experiment [24].

Fig. 2 illustrates an overview of the SDWMN control plane. The main goal of the control plane is to improve the utilization of shared bandwidth by redirecting guest traffic to one of the home routers through the WMN, based on user sharing policies. As such, the control plane exposes an interface to each home user, allowing him to express a sharing policy (e.g., using the sharing policy expression language presented in [54]). The control plane functionality mainly consists of traffic redirection and monitoring. The traffic redirection module selects the gateway for Internet access (using the algorithm presented in Section 3) and subsequently creates a tunnel between the home router where the guest is connected and the router assigned as the gateway (home routers are deployed in the AWMN RDs). Traffic is redirected through a tunnel, since the wireless nodes in AWMN are not under our control (as opposed to the RDs). A tunnel is dynamically set up when traffic has to be redirected over that path, and it is torn down when it is no longer needed. The monitoring module collects statistics about the bandwidth utilization of the WMN and the access links. A detailed description of the implementation of the SDWMN control plane and the gateway is given in Section 4.

3. Traffic redirection

Assigning flows to network paths with constrained capacity can be formulated as a multi-commodity flow problem which is known to be NP-hard [33]. Hereby, we present a heuristic algorithm for the assignment of the gateway and the path over which the traffic will be redirected through the WMN. The algorithm aims at maximizing Internet shared bandwidth utilization by accommodating as large as possible number of flows. It is executed by a SDN controller which has knowledge of the WMN topology and utilization, the Internet access link utilizations, and the user sharing policies. We first describe the traffic redirection algorithm (Section 3.1) and then investigate the potential benefits of user sharing policies (Section 3.2).

3.1. Traffic redirection algorithm

We represent the WMN as a weighted undirected graph \( G = (N, L) \), where \( N \) is the set of nodes and \( L \) is the set of links...
between nodes of the set $N$. Each node $n_i$ is associated with an Internet access link whose available shared bandwidth is denoted by $C(n_i)$. Each link $l_{ij} \in L$ between two nodes $n_i$ and $n_j$ is associated with the available bandwidth $C(l_{ij})$. Let $P_{ij}$ denote the set of paths in the network $G$, between the pair of nodes $n_i$ and $n_j$. The available bandwidth $C(p)$ associated to a path $p \in P_{ij}$ is given by the minimum residual bandwidth of the links along the path:

$$C(p) = \min_{l_{ij} \in p} C(l_{ij}) \quad (1)$$

We further represent a flow demand for a guest with the tuple $d = (n_u, r)$, where $n_u \in N$ is the node where the guest device has been attached and $r$ denotes the flow rate. We use $D$ to represent the set of flow demands that have arrived in the network.

Algorithm 1 assigns a gateway and the shortest path to each flow demand from the set $D$. The algorithm is executed whenever there is insufficient shared bandwidth in the local home router. Initially, the flow demands are sorted in decreasing order. For each flow demand, the algorithm selects the home router with the highest access link bandwidth (i.e., $C(n_u)$). Subsequently, the algorithm identifies the set of paths, $P_{ug}$, between the local home router ($n_u$) and the assigned gateway ($n_g$) that satisfy the flow demand (Eq. (1)). In case there is no such path, the algorithm performs another iteration for the selection of the gateway, excluding the previously selected home router. Otherwise, the shortest among these paths is being identified based on the number of hops (SP function in the algorithm). This eventually computes the path for the traffic redirection through the WMN.

This algorithm carries out the gateway assignment based on the available shared bandwidth of the home routers in the crowd-shared network. Although this can pool resources from all home networks where sharing is permitted achieving efficient utilization of the shared bandwidth, guest traffic may have to traverse multiple hops in the WMN. This can lead to latency inflation, delay variation, reordering [56,57] and in general, performance degradation. To mitigate this, we further use a variant of this algorithm by introducing a threshold in the number of hops between the local home router and the assigned gateway. For example, adjusting the threshold to 1 restricts the search space to the next-hop routers with sufficient shared bandwidth.

### 3.2. User sharing policies

Algorithm 1 assigns flows to gateways taking into account the amount of shared bandwidth and optionally the path hop-count. So far, we have not exploited any advance knowledge of user sharing policies for traffic redirection. Such information can prevent wasteful traffic redirections (i.e., forwarding traffic to a router which will shortly become unavailable for Internet access sharing). Sharing policies can be specified using a sharing policy expression language (SPEL), e.g., the XML-based schema presented in our previous work [54]. SPEL essentially provides the necessary means to the sharers to specify the amount of shared bandwidth as well as the period of sharing. This information is conveyed to the SDN control plane and is used as an additional input for gateway assignment.

We run a trace-driven simulation of a crowd-shared WMN to investigate the impact of sharing policies on traffic redirections. In particular, we model the availability of the home routers using an on-off Markov chain. On and off times are exponentially distributed with mean values $\mu_{on} = 106$ min and $\mu_{off} = 555$ min. We parameterize the exponential distributions using datasets from the PAWS deployment in UK.

Fig. 3 illustrates the number of traffic redirections for different network loads with and without the knowledge of sharing policies. When sharing policies are known, the traffic
is directed to the router with the longest period of availability (among the routers with sufficient shared bandwidth). According to Fig. 3, using sharing policies leads to fewer traffic redirections, especially for low and moderate network loads. For higher loads, the number of routers with sufficient shared bandwidth diminishes, and as such, the knowledge of sharing policies does not yield significant gains.

We perform additional tests to investigate potential implications, such as packet reordering and higher communication overhead, due to traffic redirections. First, we use the experimental setup depicted in Fig. 4 to quantify the level of packet reordering. In this setup, UDP traffic is redirected from path 1 to path 2 at the intermediate node. The redirection is carried out by inserting flow entries to an OpenvSwitch (OVS) datapath that has been set up in the intermediate node. We further employ a delay node, set up in path 2, to adjust the delay along this path. Fig. 5 shows that the number of out-of-order packets increases in proportion to the delay difference between the two paths. Our path delay measurements in AWMN indicate that a redirection through the WMN increases the RTT with up to 50 ms. This will result in the out-of-order delivery of a small number of packets, which, in turn, can lead to congestion control invocations, and, hence, lower throughput for TCP flows.

Besides packet reordering, traffic redirections generate additional communication overhead between the SDWMN controller and the routers. Specifically, for each flow redirection the router sends a OFPT_PACKET_IN message to the controller, which, in turn, sends an OFPT_FLOW_MOD message to the router. Taking the respective acknowledgment messages into account, the communication overhead per flow redirection is 490 bytes. This overhead can increase substantially with a large number of guest flows per home router, as the PAWS router logs indicate. Considering the identified implications of traffic redirections, we use sharing policies in the assignment of guest flows to gateways.

4. Implementation

We provide a detailed description of the implementation of the SDWMN control plane and gateway in Sections 4.1 and 4.2, respectively.

4.1. SDWMN controller

We implement our controller in Python using POX which provides a framework to implement SDN controllers with OpenFlow protocol interface [43]. Since OpenFlow supports only the configuration of switch flow tables, we use XML-RPC interface to install and configure tunnels between the gateways. Our SDWMN controller consists of the following modules (Fig. 6):

- **Gateway registration**: Each new gateway joining the crowd-shared network is registered with the creation of a new gateway object in the controller repository. Each gateway object holds information about the gateway’s:
  - (i) datapath ID, which is a 64 bit unique identifier for each OpenFlow switch instance,
  - (ii) IP address of the WMN port,
  - (iii) port table, which contains the names and OpenFlow switch ports numbers (i.e., DSL port, the WMN port and the guests’ wireless network port),
  - (iv) monitoring table, which contains the gateway shared bandwidth utilization and WMN path quality, and
  - (v) tunnel table, which contains the IP addresses, UDP port numbers (we use UDP-in-IP tunnelling) and OpenFlow switch port numbers of the installed tunnels. The registration process is triggered by POX when the gateway establishes a new control channel with the controller.

- **Monitoring**: This module collects and processes the measurements of shared bandwidth utilization and the WMN paths delay and hop counts for each gateway. Bandwidth (BW) estimation in a WMN is very challenging, due to BW fluctuations and the low degree of accuracy of available BW measurement tools (e.g., pathload [39], TOPP [44], and IGI/PTR [37]) that was also confirmed by our own tests. However, in our experimental setup, the Internet access links are the bottleneck (since they have been configured at lower capacity than the WMN), and, as such,
the aforementioned WMN bandwidth estimation issues do not affect the accuracy of our measurements. Using the OpenFlow protocol, the BW measurement module pulls the network port counters of each home gateway to acquire the accumulated number of bytes received/sent in the network port (e.g., the DSL port). In particular, the controller sends an OpenFlow `OFPT_STATS_REQUEST` message with `OFPST_PORT` type and the port number to the gateway. The gateway replies with `OFPT_STATS_REPLY` message which carries the latest reads of the gateway `rx_bytes` and `tx_bytes` counters. To calculate the shared bandwidth utilization, the BW measurement module applies exponential moving average to the counters changes over time. In addition, the path measurement module uses XMP-RPC to pull the RTT and hop-count measurements of each WMN path to/from each gateway. All measurements are stored in the monitoring table of the respective gateway object.

- **Traffic redirection**: Flows are redirected using the following modules:
  - **Gateway selection**: The gateway selection module is invoked when a gateway sends a `OFPT_PACKET_IN` message to the controller to announce the arrival of a new flow. Based on the measurements (shared BW utilization and path length) provided by the monitoring module and using the algorithm described in Section 3, the gateway selection module assigns a gateway to the new flow. Subsequently, a tunnel is being set up (see tunnel installation for further details) between the home gateway (where the flow arrives) and the assigned gateway (where the flow is redirected). Next, the flow table configuration module installs the respective forwarding entries in the home and the assigned gateway such that the guest flow is redirected accordingly.
  - **Tunnel installation**: Using the XML-RPC interface, packet encapsulation modules are installed in the gateway. This consists in sending the IP address of the assigned gateway to the flow home gateway as well as the source and destination port numbers of the UDP header used for encapsulation. Upon successful installation of the tunneling module, the home gateway replies with the switch port number of the new encapsulation module. This information is stored in the tunnel table of the home gateway object to be used later by the flow table configuration module.
  - **Flow table configuration**: This module installs the corresponding flow entries in the home gateway and the assigned gateway using OpenFlow. This consists in sending a `OFPT_FLOW_MOD` message to the home and assigned gateways. This message carries the flow matching fields [43] (e.g., source/destination IP, port source/destination) as well as the OpenFlow switch output port number.

### 4.2. SDWMN gateway

Our SDN gateway exposes to the SDWMN controller an OpenFlow and XML-RPC interface to install tunnels and redirect flows (see Fig. 7). Tunnels are created by encapsulating guest traffic in UDP packets (i.e., IP-in-UDP) using the encapsulation module (red line Fig. 7). The destination IP address of the tunnel header is set to the IP address of the assigned gateway. For each new tunnel a new encapsulation module is created. On the other hand, incoming traffic (green line Fig. 7) is processed by the decapsulation module, which strips the

![Fig. 7. SDWMN gateway. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)](image-url)
outer header before the traffic is delivered to the output port. Both encapsulation and decapsulation modules are implemented using Click [41]. To steer data traffic between the physical ports and the modules (encapsulation/decapsulation), we rely on OpenvSwitch [49]. This switching module also forwards messages from the SDN controller to the gateway controller module through the XML-RPC server. To connect modules to OpenvSwitch, we use Linux TAP devices.

The instantiation of encapsulation/decapsulation modules is performed by the gateway controller in the following steps: (i) creation of TAP interfaces, (ii) appending TAP interfaces names, MAC and IP address of the assigned gateway to Click configuration templates stored on the repository (we use templates to speed up modules instantiation) and (iii) installation of the Click configuration. Furthermore, the gateway controller collects and transfers the measurements generated by the path measurement module to the SDN controller. The path measurement module uses ping and traceroute to collect RTT and hop counts of the WMN paths. A list of gateways IP addresses is passed to this module to obtain the measurements. The module sends 10 probing packets every 30 s to avoid flooding the network.

5. Evaluation

In this section, we perform a comparative study of crowd-shared WMN against PAWS (or any other crowd-shared network with a single point of access). To this end, we run experiments in AWMN (Section 5.1) and further use simulations with larger WMN topologies (Section 5.2).

Using experimentation and simulations, we initially measure the utilization level of the shared BW and the accumulated serving rate across time and for different flow arrival rates. To study the scalability of our control plane, using experimentation we quantify the control communication overhead in terms of BW consumption and flow setup delay. We further use simulations to study the effect of traffic redirection on latency by measuring the shared BW utilization when applying a threshold to the redirection path length (see Section 3). Finally, we investigate the scalability of the crowd-shared WMN in terms of BW utilization and serving rate with different network sizes using simulations.

5.1. Experimental results

For our experiments, we use 12 research devices (RDs) to deploy 11 home routers (or gateways) and one SDWMN controller. Fig. 8 illustrates the WMN topology measured from a research device (blue circle) toward the other research devices (red circles). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

controller to announce the beginning of an on or off period. The time required to generate the signal is negligible (a few milliseconds). This timer can be also used by the sharers for sharing policy configurations in their home routers. Due to occasional reliability issues in the RDs, we restricted the duration of our experiments to 250 s. After downsampling the router on/off periods, there are several transitions between the router states during each experiment.

Our main evaluation metrics include: (i) the shared bandwidth utilization across all home routers, and (ii) the accumulated serving rate ASR(T) at time T, defined as:

\[ \text{ASR}(T) = \frac{\sum_{t=1}^{T} s_f \text{finished}}{\sum_{t=1}^{T} s_f \text{rejected}} \]

where \( s_f \text{finished} \) denotes the total sizes of the flows at time \( t \) which are accepted (i.e., assigned Internet access) and successfully served without disruption due to router unavailability. \( s_f \text{rejected} \) denotes the total sizes of the flows at time \( t \) which are rejected (i.e., not assigned Internet access).

We initially measure the shared bandwidth utilization (Fig. 9) and the serving rate (Fig. 10) with an arrival rate of 60 flows per minute, across 10 runs. Variations in the measurements across the different runs were insignificant. Fig. 9 illustrates a low utilization of the shared bandwidth without a WMN during the whole period, although there is high demand for Internet access by guests attached to various home networks. In contrast, a WMN allows to capitalize the unused capacity and accommodate a larger volume of guest traffic. More precisely, according to Fig. 9 guest traffic...
redirection through the WMN results in the full utilization of the shared bandwidth. Furthermore, crowd-shared WMNs can accommodate substantially larger volume of guest traffic, as depicted in Fig. 10. This stems from the high utilization of the shared bandwidth.

We further measure the shared bandwidth utilization and serving rate with a wide range of guest traffic demands. In this respect, the boxplots in Figs. 11 and 12 illustrate the shared bandwidth utilization and serving rate with diverse flow arrival rates, ranging from 20 to 80 flows per minute.1 These results corroborate the efficiency of the WMN for various traffic loads, as the shared bandwidth utilization and the serving rate always remain very high. In Fig. 12, the flow arrival rates of 60 flows per minute and beyond lead to request rejections, due to the insufficient access link bandwidth. On the other hand, Figs. 11 and 12 show poor bandwidth utilization and serving rate without a WMN, due to the presence of a single point of Internet access for each guest. Essentially, our results show the significant benefit that a WMN can bring into crowd-shared networks, by effectively pooling resources across all home networks.

To evaluate the scalability of our control plane, we measure the flow setup time and the average control communication overhead across a range of flow arrival rates. We define the setup time as the interval between the flow’s first packet arrival at the gateway and its transmission to the next hop. This is essentially the time incurred for gateway selection and flow entry installation. We also define the control communication overhead as the bandwidth consumed to setup flows and monitor the shared bandwidth utilization. We run

1 All boxplots show measurements across the time period of the experiment, i.e., each point on the graph represents a different time point. For different runs we calculate the average, the variation is insignificant.
our experiments on a single gateway (the results can be easily extrapolated for more gateways) and generate 100 different flows for each arrival rate. Figs. 13 and 14 show that per flow setup time does not change significantly, while the control communication overhead increases linearly and is in the range of tenths of Kbps. According to these results, the performance of our controller scales with increasing control loads.

5.2. Simulation results

To run tests at larger scale, we developed a simulator (in Python) that models the flow-level behavior of the guest traffic in a crowd-shared WMN. The flow parameters that we use in our simulations are the average flow rate, and their arrival and departure time. We use the TFA wireless mesh topology [7] which consists of 21 nodes (Fig. 15). TFA is an operational network deployed in a densely populated residential area; that is exactly the network environment we consider for the proposed crowd-shared WMN. Each node in this topology represents a home router with shared Internet access bandwidth, whereas each edge is a wireless mesh link. Guest flows arrive randomly at different home routers. Each generated flow has a rate and lifetime sampled out of a uniform and exponential distribution, respectively. The guest flows arrive to the network according to a Poisson process. We simulate the router availability using our model with the mean values obtained from the PAWS datasets. Guest flows are granted Internet access either through local routers or by redirection to remote routers based on the algorithm presented in Section 3. Flows which cannot be accommodated are rejected.

Each simulation run comprises 60 h at a time discretization of 20 min. For each scenario, we perform 20 simulation runs. We assume a homogeneous setting where each router has 16 Mbps ADSL downlink and set the shared bandwidth per router to 8 Mbps (i.e., in the periods that each router is active). We run our simulation with mesh link capacity of 200, 54 and 10 Mbps. This does not have any impact on the simulation results, since the bottleneck is the Internet access links. Note that our wireless link model does not consider the impact of interference and distance on the offered link bandwidth.

The simulation results are in accordance with the results from the community network. According to Figs. 16 and 17, the crowd-shared WMN yields significantly higher utilization of the shared bandwidth and accommodates more traffic in comparison to the crowd-shared network without WMN.
The serving rate, in particular, drops as guest flows are not granted Internet access, due to either insufficient bandwidth (with WMN) or changes in sharing policies (without WMN). Eventually, in both cases, the serving rate reaches a steady state. Shared bandwidth utilization for a wide range of guest traffic demands is depicted in Fig. 18.

In our simulations, redirected guest traffic traverses up to 4 hops in the WMN. We further use a variant of our gateway assignment algorithm (briefly discussed in Section 3), which introduces a threshold in the number of hops between the home router and the assigned gateway. Fig. 19 illustrates the shared bandwidth utilization for diverse hop-count threshold values ranging from 1 to 3, and without any limit in the hop count. Restricting the number of hops has a noticeable impact on bandwidth utilization, especially for a single hop, since Internet access is permitted only through one of the next-hop home routers. In essence, there is a trade-off between coverage extension (and thus effective bandwidth utilization) and latency. This may become more critical in large WMNs, where long paths can inflate latency.

We further run simulations using synthetic WMN topologies of diverse size and structure generated using NPART [45]. NPART is a tool for generating realistic WMN topologies, based on real-world WMN deployments (Berlin Freifunk[6]). Using NPART and the Berlin WMN model, we randomly generate 4 additional WMN topologies with 50, 100, 150 and 200 nodes. We set the wireless link capacity and shared bandwidth to 54 Mbps and 8 Mbps, respectively. For all topologies, we generate flows with arrival rate of 120 flows per minute, which allows us to saturate the WMN capacity with all topologies. As shown in Figs. 20 and 21, the shared bandwidth utilization and serving rate scale linearly with the increase of the WMN size. This is further confirmed by the increase in WMN utilization (Fig. 22) and the number of re-assigned flows (Fig. 23) with larger WMN sizes.

6. Related work

In the following, we discuss related work on WMN routing, software-defined wireless networks, and Internet access sharing.
Routing in WMN. There is a large body of work on routing in WMN [20,21,25]. This diversity stems from the different optimization goals of routing protocols (e.g., low response time, low control overhead, scalability, QoS support). More specifically, WMN routing protocols can be classified into: (i) proactive (e.g., OLSR [27], WRP [46], DSDV [48], B.A.T.M.A.N. [38]) where routes are computed in advance leading to lower response time at the cost of high routing control overhead, (ii) reactive (e.g., AODV [47], DSR) which provide routes on demand leading to lower control overhead, but higher response time, and (iii) hybrid (e.g., LOQR [32] and SrcRR [19]) where both approaches (i.e., proactive and reactive) are combined to adjust the tradeoff between response time and control overhead.

Over the years, WMN routing protocols (e.g., AODV-ST, SrcRR, B.A.T.M.A.N.) have evolved to target particular challenges or structures of WMNs. In particular, AODV-ST [50] performs routing assuming a tree-based traffic flow, where a gateway residing at the root of the tree (Internet access gateway) frequently requests routes to other nodes of the network. SrcRR computes routes taking into account the effect of interference on links quality. B.A.T.M.A.N. is another WMN routing protocol developed within the Freifunk community [6] to address the scalability limitations of the OLSR protocol and account for the static nature of WMNs.

All these protocols can be integrated into our SDN architecture for the selection of WMN paths between the home routers or as a fallback in case the connection between a router and its controller is lost. In our experimental evaluation, we rely on the WMN for the computation of routes between the local and the remote gateway (i.e., since the WMN nodes do not support OpenFlow), while we use our algorithm for gateway selection whenever traffic redirection is needed. Our approach of coupling WMN routing protocols with SDN is also employed by authors in [28,29,62].

Software-defined wireless networks. Recent work has been leveraging on SDN for WMN management. Hasan et al. [36] discuss the benefits of using SDN to decouple the operation and management of rural wireless networks from the physical infrastructure. Our work is in the same direction, but we focus on the management of crowd-shared WMN in residential areas and investigate techniques for the efficient utilization of the shared bandwidth. Authors in [28,29,62] present OpenFlow-based architectures for WMNs with emphasis on mobility management, routing, and load balancing. Dimogerontakis et al. [30] propose a SDN extension to WMN community networks for L2 experimentation. Other applications of SDN to the wireless network domain include OpenRoads [63] and OpenSDWN [59], which propose solutions for mobility management using OpenFlow. These techniques are complimentary to our work and can be integrated into our SDN control plane to facilitate mobility management in crowd-shared WMNs. Furthermore, authors in [52] investigate and evaluate the SDN control plane distribution for WMNs with emphasis on master controller selection. This can complement our work to provide higher resilience and SDN controller availability.

Internet access sharing. FON [5], Kabel Deutschland (KD) [17], Deutsche Telekom (DT) [16] and British Telecom (BT) [8] are among the network operators that provide free Internet access to their subscribers via a large number
of shared hotspots. For example, BT and KD use Home Hub and homespot, respectively, for Internet access sharing with other subscribers. Furthermore, DT and BT have partnered with FON to provide Internet access through shared home broadband connections. These efforts have also been supported by several affordable, easy-to-install and open-firmware wireless routers offered by manufacturers such as FON [9], openMesh [15], Linksys [13], and Netgear [14].

Our approach leverages on SDN to bring the following benefits to Internet access sharing: (i) reduction of operational expenses for home networks and ISPs, since the deployment, configuration, and operation of crowd-shared networks can be outsourced to third-parties (i.e., virtual network operators), and (ii) better coordination of network configurations for traffic redirections, due to the global WMN view and the collection of all monitoring and sharing policy information in a centralized controller.

In addition, Link-alike [40] promotes sharing by aggregating the uplink bandwidth of the broadband connections through WMN, such that home users have access to higher upstream capacity. On the other hand, ARBOR [61] aggregates broadband connection bandwidth (both uplink and downlink) from multiple WiFi networks. We achieve the same goal (i.e., high utilization of the shared bandwidth) via guest traffic redirections through the WMN, enabling resource pooling from all shared broadband connections.

7. Conclusions

In this paper, we investigated the benefits of extending the coverage of any crowd-shared network (e.g., PAWS) by connecting the home routers as a mesh. A crowd-shared WMN can mitigate the fundamental problem of any crowd-shared network, i.e., the presence of a single point of access for each guest. We showed that the advance knowledge of user sharing policies can reduce the number of guest flow redirections (particularly for low and moderate network loads) avoiding implications, such as packet reordering and control communication overhead. This leads to a significantly higher utilization of the shared bandwidth, as opposed to a crowd-shared network with a single point of access where a portion of the shared bandwidth is wasted. This, in turn, enables a software-defined crowd-shared WMN to accommodate a substantially larger volume of guest traffic.

SDN brings significant benefits to crowd-shared networks, as all sharing policy and WMN utilization information can be conveyed to a centralized controller facilitating the assignment of guest flows to gateways and the traffic redirection configurations in the WMN. In contrast, a decentralized crowd-shared WMN management would require synchronization primitives and could yield slow convergence, especially at the event of message loss. Our experimental results show that our SDWMN control plane exhibits low per-flow setup time and low control communication overhead.

Deploying a SDWMN in the real world raises challenges, such as the controller placement, control plane distribution for robustness and load balancing, and SDN support SDN in home routers. We envision hosting the SDN controller as a virtual machine in micro-datacenters hosted by large ISPs [35]. Micro-data centers are small DCs deployed at POP and network aggregation points. By deploying the controller in micro-DCs we ensure low communication delay with the home routers and scalable processing power to cope with high loads (scale up the controller computation capacity by using more servers). We can further leverage on several frameworks to distribute the control load across multiple servers, leading to faster response time and providing a fallback controller in case of failure [22, 23, 31, 42, 52].

In terms of SDN support in home networks, there are already OpenFlow-based wireless routers (e.g., HP [11], Meru [12]). Furthermore, OpenFlow has been ported to OpenWRT [9, 10], allowing wireless routers running OpenWRT to be configured using OpenFlow. This essentially facilitates the extension of PAWS as a crowd-shared SDWMN, since PAWS already uses wireless routers with OpenWRT-based firmware.

Content caching and the ever-growing penetration of online social networks (OSN) create for more opportunities in software-defined crowd-shared networks. More specifically, popular content can be placed in caches within home networks, while the SDN controller can redirect guests to caches, conserving bandwidth in shared broadband connections. Furthermore, information can be utilized from OSNs in order to associate and authenticate guests to access points that belong to their online friends, leveraging on the trust between two parties with an OSN relationship [26]. In future work, we plan to investigate any potential benefits by coupling content caching and OSNs with crowd-shared WMNs.

Acknowledgments

This work was partially supported by the EU CONFINE Project (288535). We thank Amr Rizk for his help in modeling the home router on/off periods.

References

Panagiotis Papadimitriou is an Assistant Professor at the Communications Technology Institute of Leibniz Universität Hannover. He received a Ph.D. in Electrical and Computer Engineering from Democritus University of Thrace, Greece, in 2008. From 2008 to 2010, Panagiotis was a Post-Doctoral Researcher at the Computing Department of Lancaster University, UK, and at Telecom SudParis, France. Panagiotis received the runner-up Poster Award at ACM SIGCOMM 2009 and the Best Paper Award at IFIP WWIC 2012. The research activities of his group currently span next-generation Internet architectures, network function virtualization, software-defined networks, and cloud networking.

Arjuna Sathiaseelan is a Senior Research Associate at the Computer Laboratory, University of Cambridge. He leads the Networking for Development (N4D Lab). The research group conducts research on novel Internet architectures for improving and reducing the cost of Internet access. He is the Chair of IRTF Global Access to the Internet for All (GAIA) research group and a member of the Internet Research Steering Group (IRSG). He has a Ph.D. in Networking from Kings College London (2005), M.Sc. in Computing and Internet Systems from Kings College London (2001) and Bachelors in Computer Science and Engineering from NIT, Trichy, India (2000).