The SMesh Wireless Mesh Network

Yair Amir[1], Claudiu Danilov[2], Raluca Musăloiu-Elefteri[1], Nilo Rivera[1]
[1] Johns Hopkins University
[2] Boeing Phantom Works

Technical Report CNDS-2009-3 - April 2009
http://www.dsn.jhu.edu

Abstract

Wireless mesh networks extend the connectivity range of mobile devices by using multiple access points, some of them connected to the Internet, to create a mesh topology and forward packets over multiple wireless hops. However, the quality of service provided by the mesh is impaired by the delays and disconnections caused by handoffs, as clients move within the area covered by multiple access points. We present the architecture and protocols of SMesh, the first transparent wireless mesh system that offers seamless, fast handoff, supporting real-time applications such as interactive VoIP. The handoff and routing logic is done solely by the access points, and therefore connectivity is attainable by any 802.11 device. In SMesh, the entire mesh network is seen by the mobile clients as a single, omnipresent access point, giving the mobile clients the illusion that they are stationary. We use multicast for access points coordination and, during handoff transitions, we use more than one access point to handle the moving client. SMesh provides a hybrid routing protocol that optimizes routes over wireless and wired links in a multi-homed environment. Experimental results on a fully deployed mesh network demonstrate the effectiveness of the SMesh architecture and its intra-domain and inter-domain handoff protocols.

1 Introduction

Wireless networks have changed the way people connect to the Internet, giving users the freedom to connect from anywhere within the coverage are of a wireless access point. Wireless Mesh Networks extend the wireless coverage area of an access point by having a few access points connected to a wired network, and allowing the others to forward packets over multiple wireless hops. A mesh network can span a large geographical area. Internet connected access points (*Internet gateways*) may reside at different network domains, effectively creating a *multi-homed* wireless mesh network.

When a user moves outside the range of an access point and closer to another, it switches its connectivity to the closer access point. This connectivity change involves a transition (*handoff*) before being able to route packets to and from the new access point. Maintaining connectivity requires a handoff at two levels. An intra-domain handoff is required to transfer connectivity between the access points serving the mobile device. At a higher level, an inter-domain handoff between access points connected to the Internet may be required on existing connections. Both handoffs, which can occur simultaneously, must maintain all previously opened connections while transferring them as fast as possible. Ideally, the handoff should be completely transparent to mobile clients. There should be no interruption in network connectivity, and the communication protocols involved should follow the standards deployed in regular wireless devices. We call a wireless network that offers such a service a *seamless* wireless mesh network.

While cell phone networks solve the handoff problem using signaling embedded in their low-level protocols [9, 17], there are currently no efficient, transparent handoff solutions for wireless 802.11 networks. Most wireless networks today require specially modified clients in order to transfer connectivity from one access point to the next. Others, even if they give the appearance of continuous connectivity to a roaming client, provide connections that are in fact interrupted when a client transfers from one access point to the next, with delays that can be as long as several seconds [33, 48]. For some applications (e.g., transferring files), this delay may be acceptable (although it will lower the overall throughput). However, it is far too long for real-time traffic such as interactive Voice over IP or video conferencing.

This paper presents the architecture and protocols of SMesh [6, 7], the first transparent wireless mesh network that offers seamless fast handoff, supporting VoIP and other real-time application traffic. The entire handoff and routing logic is done solely by the access points, and therefore connectivity is attainable by any 802.11 mobile device, regardless of its vendor or architecture. In order to provide this level of transparency to mobile clients, our approach uses only standard network protocols. The entire mesh network is seen by the mobile clients as a single, omnipresent access point, giving the mobile clients the illusion that they are stationary.

Mobile clients are handled by a single access point during stable connectivity times. Fast intra-domain handoff is achieved by controlling the handoff from the mesh infrastructure and by using multicast in the mesh network to send data through multiple paths to the mobile client during handoff. Access points continuously monitor the connectivity quality of any client in their vicinity and efficiently share this information with other access points in the vicinity of that client to coordinate which of them should serve the client. If multiple access points believe they have the best connectivity to a mobile client, and until they synchronize on which should be the one to handle that client, data packets from the Internet gateway (or another source within the mesh network) to the client are duplicated by the system in the client's vicinity.

Fast inter-domain handoff is achieved by using multicast groups through the wired network to coordinate decisions and seamlessly transfer connections between Internet gateways as mobile clients move between access points. New connections always use the closest Internet gateway at the time of their creation, while existing connections are forwarded through the wired infrastructure to the Internet gateway where they were originally initiated. As the handoff process requires routing agreement and transferring connections between the involved Internet gateways, our protocol guarantees that packets are routed correctly, at all times.

While duplicating packets and tightly coordinating the access points in a client's vicinity may seem to incur high overhead, this paper quantifies the overhead and demonstrates it is negligible compared to data traffic.

We also show how our system optimizes peer-to-peer communication between mobile clients, utilizing the wired connectivity available at the Internet gateways to reduce wireless usage. The forwarding and coordination between the access points is done using our Spines messaging system [43] that provides efficient unicast, anycast, and multicast communication in an overlay network environment.

The contributions of this paper are:

1. The architecture and protocols of the first seamless wireless mesh network with fast intra-domain and inter-domain handoffs that support real-time applications such as interactive VoIP and video conferencing.

2. Novel use of multicast for robust mesh Internet gateway to client communication, as well as for access point coordination.

3. Novel use of anycast for mobile client to mesh Internet gateway communication.

4. A hybrid routing protocol for mesh communication that optimizes routes over wireless and wired links in a multi-homed environment.

5. A set of experiments in a real-world deployment demonstrating the effectiveness of the SMesh system.

The rest of the paper is organized as follows: The next section overviews related work, followed in Section 3 by a description of a generic wireless mesh network environment. Section 4 presents the architecture of our wireless mesh system, SMesh. Section 5 presents our fast intra-domain handoff protocol, which includes client monitoring, mobility management, and the fast handoff approach. In Section 6, we present our fast inter-domain handoff for multi-homed wireless mesh networks, showing how TCP and UDP connections are separately handled to correctly route these packets. We present experimental results in Section 7, and Section 8 summarizes our contribution and concludes the paper.

2 Related Work

Seamless mobility in wireless mesh networks must account for movement at two different levels: intra-domain, between access points, and inter-domain, between Internet connected access points potentially connected on different networks. As such, our work relates to previous work on wireless mesh networks, intra-domain handoff, and inter-domain handoff. In addition, our work relates to overlay networks, as the underlying communication infrastructure of SMesh relies on an overlay messaging system.

Good surveys addressing most of these areas are provided by [3] and [2]. Note that related work may also refer to intra-domain handoff as *micromobility* and to inter-domain handoff as a form of *macromobility*.

2.1 Wireless Mesh Networks

One of the first commercial mesh networks was Metricom's Ricochet network [44] in the mid-90s. Ricochet nodes automatically routed client traffic through half-duplex wireless hops until reaching a hardline connection.

When the 802.11 standard was ratified in the late-90s, other mesh networks started to emerge. One of these is the MIT Roofnet project [16], [10] in which tens of access points with roof mounted antennas form a mesh around campus. Roofnet's emphasis is more on route maintainability and optimization than on handing off a client's connection. Many other community and commercial mesh network implementations also exist, such as Rice University TAPS in Houston [14] and Urbana-Champaign Community Wireless Project [20].

Microsoft Research has also done notable work in the area of mesh networks. Their Mesh Connectivity Layer (MCL) [23] creates a wireless mesh network between Windows clients. Their focus is on efficient routing protocols [24] along with the unique support for multiple radios on each node [1]. MCL requires a specific network driver on all mesh network participants, including the clients.

[32] studied the throughput capacity of hybrid networks that connect some of the nodes through the wired network to improve efficiency in the use of the wireless spectrum. In our routing strategy we also take advantage of the wired connections available at the gateways.

The IEEE 802.11s Mesh Networking standard, analyzed in [13], specifies three different types of mesh nodes. Mesh points (MP) includes all mesh nodes that participate in the wireless backbone to increase the mesh connectivity. Some mesh points serve as mesh access points (MAP), providing connectivity to clients within their wireless coverage area. Also, some mesh nodes may serve as mesh portals (MPP), connecting the wireless mesh to an external network such as the Internet. In our approach, we assume that every node is potentially an access point, as it increases the availability of the system. Furthermore, other than Internet connectivity, we make no distinction between the capabilities available in nodes that are simply MAP, MPP, or both.

2.2 Intra-domain Handoff

Cell networks achieve smooth handoff by sharing information between towers about a given mobile device. This session data is used for routing and is updated whenever a phone switches cells [9], [17]. The 802.11 standard lacks the handoff mechanisms available in today's cell network protocols.

[33] analyzed the link-level handoff performance in current 802.11 hardware. Approximately 90% of a handoff delay is attributable to the client adapter scanning for its next AP. Their experiments also illustrate that the practical handoff delay can vary widely depending on the vendors used for the client network card and the AP. [47] investigated the latency effects of a wireless handoff on voice traffic. The conclusions echo those of Mishra et al. in that the handoff latency can vary widely depending on the hardware vendor used. Since our approach does not require re-association during handoff, we do not suffer from these vendor specific delays.

[38] recently demonstrated that a quick link-level handoff is possible on 802.11 networks when the client monitors the signal quality of access points and uses a fast scanning mechanism to listen to all APs in range to choose the best one. Their SyncScan system has achieved an impressive handoff as low as 5 ms. The fast scanning is achieved through driver modifications of the client's network adapter. In the contrary, our approach uses any unmodified 802.11 client.

Two well known general approaches to intra-domain handoff in IP networks are Cellular IP [46] and Hawaii [39]. A comparison is presented in [15]. In Hawaii, or Handoff-Aware Wireless Access Internet Infrastructure, messages are exchanged between the old gateway and the new gateway for forwarding packets. Cellular IP establishes routes based on the traffic from the client, and handoff takes place when a cross-over router is reached. [12] use a different approach to mobility in which access points send gratuitous ARPs to their upstream routers to create the illusion that mobile clients are always connected to the wired network. These approaches rely on clients initiating the handoff process, and do not address the link level handoff delay present in 802.11 networks when clients re-associates with another access point. Other approaches to intra-domain handoff, such as TMIP [28], [49], and [41], improve handoff latency in 802.11 networks but do not overcome these limitations. Other general approaches such as IDMP [21], SMIP [31], and HMIP [29] focus on hierarchy to reduce the global signaling load and improve scalability. Most of the above require software be installed in the mobile clients. In contrast, we provide a complete link-level and network-level solution and propose a novel approach to seamlessly control the handoff from the infrastructure.

Seshan, Balakrishnan, and Katz used a multicast approach in the Daedalus project [40] to ensure timely delivery of client traffic during a handoff in a cell-based wireless computer network available in 1996. Later, [30] show how fast handoff can be achieved in wireless networks by requiring mobile clients to explicitly join a multicast group to which packets are multicast-tunneled through the infrastructure. Multicast during handoff, referred to as simulcast, is also used during handoff in S-MIP [31]. Our approach also relies on multicast during handoff, but only in the mesh, to reach multiple access points, and therefore it does not require any modifications to the mobile client, thus supporting standard mobile devices of any architecture or operating system.

The IEEE has also been working on standardizing handover for wireless IP networks at two different levels. The 802.11r standard aims at providing fast Basic Service Set (BSS) transition by allowing clients to use their current access point as a conduit to other access points. The 802.21 standard aims at providing handover between different network types, commonly known as media independent or vertical handover. These approaches require modifications to the 802.11 standard, and so to the access points and to every client device. In our approach, no modifications are necessary.

Existing experimental wireless mesh testbeds that support client mobility include MeshCluster

[37] and iMesh [45], both of which work with mobile clients in infrastructure mode. MeshCluster, which uses Mobile IP (MIP) [35] for intra-domain handoff, shows a latency of about 700 ms due to the delay incurred during access point re-association and MIP registration. iMesh also offers intra-domain handoff using regular route updates or Mobile IP. Using layer-2 handoff triggers (no moving client), handoff latency in iMesh takes 50-100 ms. The approach was later used in a more realistic environment for improving VoIP performance in mesh networks, with similar results [27]. Our approach provides 802.11 link-layer and network-layer fast handoff by working in ad-hoc (IBSS) mode, controlling handoff from the mesh infrastructure, and using multicast to send data through multiple paths to the mobile client to deal with incomplete knowledge and unpredictable moving patterns.

2.3 Inter-domain Handoff

Two general approaches for supporting inter-domain handoff are Mobile IP (MIP) [35] and Mobile NAT [11]. In MIP, a client binds to an IP address at the Home Agent (HA). As the mobile client moves to a different access point or domain, it receives a Care-of-Address (CoA) from a Foreign Agent (FA). The mobile client then registers its new CoA with its HA, and data is then tunneled through the HA. Our approach does not require binding the mobile client to a specific Home Agent, but rather ties each connection to the Internet gateway that is closest at the time the connection is initiated.

In Mobile NAT, a client receives two IP addresses through DHCP: a binding address for the network stack, and a routing address that will be visible in the network. As the mobile client moves to a different domain, the client may receive a new routing address. However, as end-to-end connections were initiated from the IP address of the network stack, which remains the same, existing connections will be maintained. The approach requires modifying the mobile client network stack to be aware of the protocol, and also changes in the standard DHCP protocol. Our approach does not require any modifications to the mobile client or the DHCP standard.

2.4 Overlay Networks

Overlay networks enable developers to implement new services on top of the IP network infrastructure without requiring special support from the underlying network. They are usually built as application level routers to ensure flexibility and usability across platforms, at the cost of requiring packet to traverse through user space. Examples of application level overlay routers include RON [8], End-System-Multicast [18], and Spines [43, 4].

RON routes packets through a user level router on an overlay network to increase the reliability of the end-to-end path when compared to using the underlying direct path. End-System-Multicast also routes through an application router to support overlay multicast without infrastructure support.

Spines is a more generic overlay network that provides transparent multi-hop unicast, multicast and anycast communication with a variety of link and end-to-end protocols. For example, semi-reliable links can recover from some loss in the overlay links while packets are independently forwarded to their destination in order to improve VoIP [5] quality. Spines has a socket-like interface that makes the interconnection with other components very easy. It uses an addressing space composed of virtual IP addresses and virtual ports. Regular socket calls such as *sendto()* or *recvfrom()* are mapped directly into Spines API calls. The SMesh system instantiates a Spines daemon on each wireless mesh node to manage group membership and to forward messages within a multi-homed wireless mesh network.

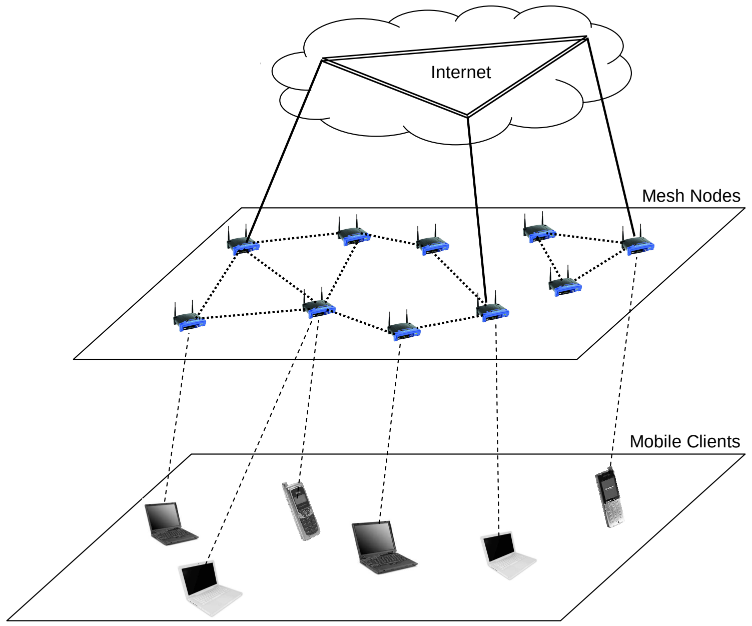


Figure 1: A Multi-homed Wireless Mesh Network.

3 Environment

A wireless mesh network is comprised of multiple access points, possibly distributed in several islands of wireless connectivity such as different buildings located close to each other or parts of the same building. Access points inside a wireless island can communicate with each other, potentially using multiple intermediate hops. We call each access point a *node* in the wireless mesh network. For Internet connectivity, one or more access points in each wireless island can directly connect to the wired network. Other access points rely on multi-hop communication to reach an Internet connected node in their island.

The mesh clients are regular 802.11 devices that communicate with the mesh nodes to get access to the network. We do not assume any specific drivers, hardware, or software present on the clients. Therefore, *any* regular unmodified mobile device should be able to use the mesh network transparently.

Figure 1 depicts a general overview of a the wireless mesh network paradigm. While the mesh nodes are usually stationary, mobile devices that connect to the mesh network can roam throughout the coverage area. This is one of the main differentiating factors between the mesh network and the Mobile Ad-hoc Network (MANET) paradigm, where everyone (mesh nodes and mesh clients) can move and participate in the overall routing protocol.

Even though the mesh nodes are stationary, the mesh topology changes when wireless connectivity between the mesh access points changes, when nodes crash or recover, or when additional nodes are added to expand the wireless coverage.

4 The SMesh Architecture

Our goal is to allow mobile clients to freely roam within the area covered by the wireless mesh nodes, with no interruption in their Internet connectivity. All connections (reliable or best effort) opened at mobile clients should not be affected as the clients move throughout the coverage area served by the wireless mesh.

Following the above goals, we implemented SMesh[42], a wireless mesh network system that provides seamless connectivity to mobile clients. The software architecture of SMesh is shown in Figure 2. Below we describe the two main components of the SMesh architecture: the communication infrastructure and the interface with mobile clients.

4.1 Communication Infrastructure

The mesh nodes create a relatively stable ad-hoc wireless network. Within this network, the nodes need to forward packets over multiple hops in order to communicate with each other for reaching the Internet gateways or for coordinating decisions about serving mobile clients. The nodes also need to discover and monitor their neighbors and to automatically adjust the mesh routing in case of topology changes.

The communication infrastructure of SMesh relies on the Spines messaging system [43, 4]. The Spines overlay network interconnects all nodes through direct links in the wireless network and through virtual links in the wired network. SMesh instantiates a Spines daemon on each wireless mesh node to forward messages within the wireless mesh. Each daemon keeps track of its own direct neighbors by sending out periodic hello messages. Based on the available connectivity, each node creates logical wireless links with its direct neighbors and uses a link-state protocol to exchange routing information with other nodes in the network.

The nodes flood link-state information using reliable links between direct neighbors. This allows the nodes to send only incremental updates, and only when network topology changes. Link state updates contain only information about the wireless links that change their status. When there are no changes in topology, no routing information is exchanged. Considering that mesh nodes (access points) are mostly stationary and that topology changes are relatively rare, the incremental link-state mechanism incurs very low overhead. Note that in SMesh, mobile clients are not part of the mesh topology. While this link-state protocol may not be optimal for a mobile ad-hoc network, it is suitable for the relatively stable network underlying our mesh of access points.

Running as a software router in user-space, Spines allows us to use multicast and anycast functionality in a multi-hop wireless environment without infrastructure support. A multicast group is defined as a class D IP multicast address while an anycast group is a class E IP address. Note that the groups are defined in the Spines virtual addressing space, not in the actual IP address space of the network. When a mesh node joins or leaves a group, the local Spines daemon informs all the other nodes in the network through a reliable flood similar to the link-state protocol. Only joins and leaves are flooded to the mesh nodes in the system. The group membership is maintained in Spines in tuples of the form (*mesh_node_address, group_address*), such that each node knows all the groups that other nodes are members of.

Based on the group membership and available connectivity, Spines automatically builds multicast trees throughout the mesh network. A multicast data message follows the multicast tree corresponding to its group. Therefore, if several nodes in a certain vicinity join a multicast group, multicast messages exchanged between them will only be sent in that vicinity. An anycast data message follows a single path in the tree to the closest member of the group.

Multicast trees in Spines are built by optimizing on a metric that can be based on number of hops, link latency or loss rate. In our tests, Spines could handle several hundred thousand group members on regular desktop machines and was limited only by the available memory to maintain the data structures. As we will show later, SMesh instantiates two groups for each client, with a few members in each group. The more limited Linksys WRT54G routers used in our experiments have enough memory to support at least 1000 mobile clients at the same time.

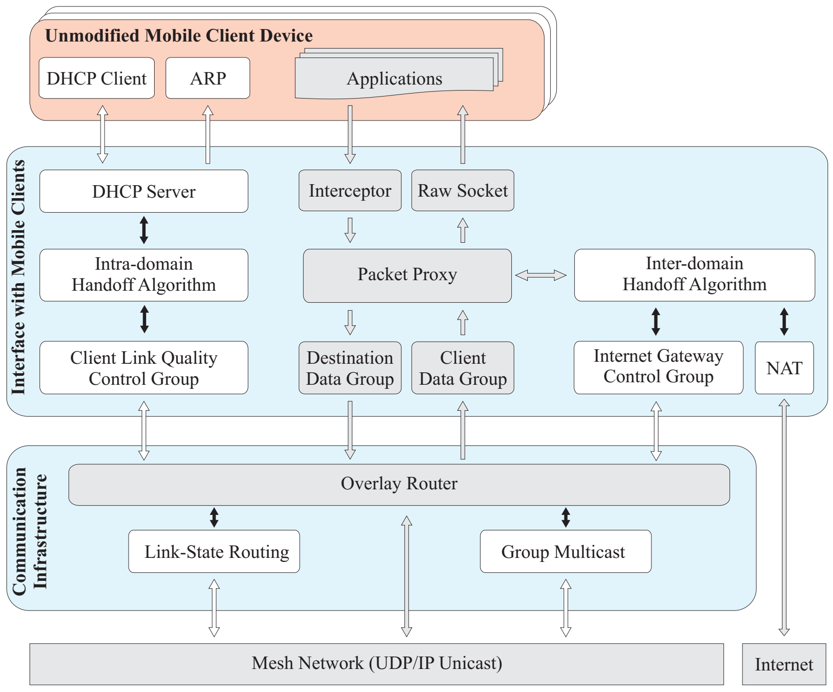

Unmodified Mobile Client Device
DHCP Client
ARP
Applications
Interface with Mobile Clients
DHCP Server
Interceptor
Raw Socket
Intra-domain Handoff Algorithm
Packet Proxy
Inter-domain Handoff Algorithm
Client Link Quality Control Group
Destination Data Group
Client Data Group
Internet Gateway Control Group
NAT
Communication Infrastructure
Overlay Router
Link-State Routing
Group Multicast
Mesh Network (UDP/IP Unicast)
Internet

Figure 2: The SMesh Architecture.

4.1.1 Topology formation

The topology formation of the mesh network starts with each access point broadcasting its presence periodically. Neighboring nodes create bidirectional links and advertise their connectivity through a link state protocol to other nodes in the network.

Internet gateways join a multicast group called *Internet Gateway Multicast Group* (IGMG) on which they periodically advertise their wired interface IP address. This multicast group is also handled by the underlying overlay infrastructure. When two Internet gateways receive each other's advertisements (which initially travels through the wireless infrastructure to the members of the multicast group), they connect through a wired overlay link. This way, the Internet gateways inside an island form a fully connected graph using their wired infrastructure, while the other access points inside the island interconnect based on the wireless connectivity. In order to interconnect wireless islands, at least one Internet gateway in each island needs to be pre-configured to connect to a set of Internet gateways such that an initial connected graph is formed. Then, multicast advertisements from all gateways will be propagated, Internet gateways will connect to each other, and eventually, a fully connected logical graph between all Internet gateways in all islands is formed.

4.1.2 Routing metric

In a mesh network with multiple Internet gateways, wired connections can be used to shortcut several hops of wireless communication, thus decreasing the number of wireless transmissions. For this reason our system maintains a hybrid overlay infrastructure, with both wired and wireless links. In general, in a combined wired-wireless routing metric scheme, it is reasonable to assume that a wired connection costs much less than a wireless link. On the other hand, depending on the network conditions it is possible that wired connections between Internet gateways have different costs (based

Type	Address	Example	Details
Client IP	10.A.B.C	10.11.12.25	Assigned by SMesh DHCP Server
Netmask	255.255.255.248	255.255.255.248	Assigned by SMesh DHCP server
Default Gateway	10.A.B.C + 1	10.11.12.26	Assigned by SMesh DHCP Server
Network Address	10.A.B.C - 1	10.11.12.24	Calculated by Client with Netmask
Broadcast Address	10.A.B.C + 6	10.11.12.31	Calculated by Client with Netmask
Reachable IP	10.A.B.C + 2	10.11.12.27	Used by SMesh to monitor the client

Table 1: SMesh IP addressing scheme.

on throughput, loss rate, latency, etc.).

Our approach uses the best route to a destination considering wireless connectivity as well as any hybrid route available, and allows for different routing metrics to be used both on the wired and wireless links. Considering that each wireless link can have an *ActualCost* metric of at least 1, the routing cost of that link will be:

$$Cost = ActualCost * (M + 1)$$

where M is the maximum cost that can be associated with a wired path. For example, if a wired link can have a maximum cost of 10, and there are 5 access points connected to the Internet in the mesh network, the value of M is 40 (the largest number of wired hops in a path is 4), and the minimum cost of a wireless link is 41. The cost of a hybrid path is the sum of the cost of all the links. This mechanism gives preference to any wired link over a wireless one, and optimizes the wired path based on a desired metric. For example, we can use ETX [19] as the wireless *ActualCost* metric, and latency as the wired links metric.

4.2 Interface with Mobile Clients

SMesh provides the illusion of a single distributed access point to mobile clients. This is achieved by providing connectivity information to clients through DHCP [25] and by always giving the same information (IP address, Netmask, and Default Gateway) to the mobile client.

4.2.1 Mobile client connectivity

Each mesh node runs a DHCP Server that is in charge of providing network bootstrap information, including a unique IP address, to a requesting client. We compute this IP address using a hash function on the client's MAC address, mapped to a class A private address of the form 10.A.B.C. A small portion of the private IP addresses in this range is reserved for SMesh nodes, and the rest are available to mobile clients. In case of a hash collision, the client with the smallest MAC keeps the current IP and any other client in the collision gets a managed IP. This scheme decreases the amount of IP management in the network, while assuring that each client gets the same IP address from any SMesh node.

Of particular importance in the DHCP protocol are the *Server ID*, *Default Gateway*, and the T_1, T_2 and *Lease* timers. The *Default Gateway* specifies the next hop router to use at the MAC level when sending to an IP address outside the client's netmask. The *Server ID* specifies the DHCP Server IP address that the client should contact to renew its lease. The T_1 and T_2 timers specify when to start unicasting or broadcasting DHCP requests (`DHCPREQUEST`), and the *Lease* timer specifies when the client must release the IP address. After the *Lease* timer expires, all the connections at the client are terminated. If the access point responds to a DHCP request before the client's Lease time expires, it is able to keep all connections open. In SMesh, the lease time is set to 90 seconds, which gives a client enough time to reconnect in case it goes out of range of any of the mesh nodes temporarily.

Table 1 shows our addressing scheme. We assign a small subnet to each client. Every client resides in a different subnet, thus forcing the client to send packets destined to the Internet or to a peer through its default gateway. The default gateway is set to a virtual IP address; there is no node in SMesh with that IP address. Instead, SMesh makes the client "believe" that this address is reachable by associating this IP address to a mesh node hardware address. This forces the client to route packets through a SMesh access point.

While each client in SMesh consumes 3 bits from the address space, there are still 21 bits available, which allows us to support over one million client IP addresses.

Section 5 explains how the virtual default gateway is mapped to a real access point, how we use the DHCP timers, and how an additional IP address in the client subnet is used to monitor its connection.

4.2.2 Packet proxy

Mesh nodes serve as default gateways for the mobile clients. A Packet Proxy module, depicted in Figure 2, uses an interceptor to grab packets from a client, and a raw socket interface to forward packets back to the client.

Each mobile client is associated with a unique multicast group in the mesh, Client Data Group, for access points to receive client data. One or more mesh nodes that are in the vicinity of a client will join that client's Data Group.

If the destination of a packet is a SMesh client, the packet is sent to the SMesh nodes that joined that client's Data Group. The mesh node sending this packet can be the Internet Gateway (for packets coming from the Internet) or a sending client access point (for packets originated by a different SMesh client). Upon receiving a packet for the client, each of the SMesh nodes that joined that client's Data Group forwards the packet to the client.

If the destination of a packet is the Internet, then the packet is sent by the originating client's access point to the closest Internet gateway by forwarding it to an anycast group that all Internet gateways join. Since clients reside in a private address space, Internet gateways perform a Network Address Translation (NAT) [26] before forwarding the packet to the Internet. When a response packet is received from the Internet, a reverse NAT is performed and the packet is sent to the appropriate Client Data Group. Under normal circumstances, only one mesh node join this group.

Spines forwards the packets to the members of the client's Data Group using a multicast tree. This way, if the mobile client moved, and a different SMesh node joins the client's Data Group, the packets are forwarded to the newly joined SMesh node. The SMesh node(s) in the Client Data Group use a raw socket to deliver the packet, allowing the mobile client to receive the packets unmodified as if it had a direct connection to the end host. If there are multiple nodes in the Client Data Group, the client could receive duplicate IP packets. However, duplicate IP packets are dropped gracefully at the receiver (TCP duplicates are dropped at the transport level, and applications using UDP are supposed to handle duplicates).

5 Fast Intra-domain Handoff Protocol

Real-time applications such as VoIP require that packets arrive in a steady stream. Any burst of loss where consecutive packets are lost results in degradation of quality. In addition, in the case of VoIP, packets should arrive within 100 ms to prevent a noticeable delay that impairs interactivity, and delay variability should stay below 20 ms to ensure the highest quality of service. Therefore, a handoff protocol should be fast enough to avoid any packet loss, and should ensure that packets are delivered to their destination in a timely manner.

When 802.11 devices are configured in *infrastructure mode* (BSS), they inherently perform their own scanning for a better access point. A layer 2 handoff takes place through a re-association request/response process which can last as long as several seconds [48]. In addition, this handoff is both hard and forward; hard because the client can only speak with one access point at a time, and forward because the client can not communicate with its old access point during the handoff process. A typical handoff will last about 500 ms, which translates to dozens of lost packets per handoff for VoIP applications.

In order to avoid this behavior and control the handoff solely from the access points, we configure both the access points and the mobile clients in *ad-hoc mode* (IBSS). This is one of the standard modes of operation available on any 802.11 compliant device.

One way to perform the handoff in ad-hoc mode is by relying on the DHCP protocol. For example, one can instruct the client to renew its lease every few seconds. Any access point that hears the DHCP request may respond and become the default gateway for the client. While this mechanism may provide some handoff capability, handoff can still take seconds as nodes need to wait for the client to initiate the DHCP transaction. Moreover, the client may connect through an access point that has a weak connection, while better nodes may be available.

Instead of letting the client "decide" when the handoff should take place, we make the SMesh nodes track their connectivity to the client and force the client to change its access point when better connectivity is available (avoiding oscillations is described below). To achieve this without modifying anything on the client side, we provide the illusion of a single IP that never changes as the default gateway of the client and use gratuitous ARP messages to force roaming to the SMesh node with the best client connectivity.

The details of our intra-domain handoff protocol are described below. These include measuring and sharing the link quality metric to determine the best access point for each client, the use of overlay multicast groups for managing the clients, and the actual handoff process.

5.1 Mobile Client Monitoring

5.1.1 Seamless heartbeat with DHCP and ARP

In order to provide continuous connectivity and availability to the mobile client, we need to continuously monitor the client. To achieve seamless monitoring without any involvement from the client, we developed two strategies.

1. DHCP (Dynamic Host Configuration Protocol)

   According to the DHCP standard [25], the T_1 (Renew) and T_2 (Rebind) timers specify when to start unicasting and broadcasting, respectively, DHCP requests (`DHCPREQUEST`), and the *Lease* timer specifies when the client must release the IP address. After the *Lease* timer expires, all the connections at the client are terminated. If the access point responds to a DHCP request before the client's Lease time expires, it is able to keep all connections open. When using the SMesh DHCP monitor, our DHCP server instructs the clients to renew their IP address every 2 seconds, thus serving as a heartbeat to keep track of the client. In addition, the timers may be set so that the client unicast or broadcast their request every 2 seconds. On the down side, it employs a non-negligible overhead as a `DHCPREQUEST` packet is at least 300 bytes long, and a `DHCPACK` is about 548 bytes. Another downside is that, when the first `DHCPREQUEST` is lost, the time between this request and the next is platform dependent and usually more than several seconds. This is the approach we took in [6].

2. ARP (Address Resolution Protocol)

   ARP [36] protocol is used to map an IP address to a hardware address (MAC), when a host (or router) wants to communicate with another host inside the same network. In general, given an IP address for which its corresponding hardware address is not present in the ARP cache of a client, the ARP module of that client will broadcast an ARP request packet. In addition to the source and destination IP addresses, this ARP request contains the MAC address of the source. The value of the destination MAC is not yet known. All the hosts on the local network receive the packet and compare the destination IP with their own IP address. The host for which the IP address matches will issue an ARP reply, filling in the destination MAC field with its own MAC address. This packet is sent directly via unicast to the requesting client.

However, even if the hardware address is known, we can still use this protocol to probe the client's link and estimate its loss rate. By using regular ARP requests, we can make the client either unicast or broadcast ARP responses. We instruct the client to send its reply to a special IP inside its subnet, with the MAC address of the access point that sent the reply (e.g., a heartbeat sent by access point 10.0.0.31 to client 10.11.12.25 is ARP packet *"Who has 10.11.12.25? Tell 10.11.12.27."*, where the MAC address associated with 10.11.12.27 is set to be the real MAC of 10.0.0.31). This is necessary as the real IP address of the SMesh nodes is outside the client network. The advantage of using this approach is that, unlike DHCP, ARP packets are very small, only 28 bytes. In SMesh, we request an ARP reply from the client every one or two seconds. Also, to limit the number of access points probing the client, only the mesh node in the client Data Group periodically sends a request, and all nodes in the vicinity use the reply to compute the metric. This is the approach that we take in our current version of SMesh and for the experiments presented in this paper.

It is also possible to use regular packets sent by the client to monitor its connectivity[1]. However the approaches presented above are still necessary when the client is idle and there is no incoming/outgoing traffic.

5.1.2 Quality metric

We use the monitoring schemes described above to keep track of the quality of the links to mobile clients. Both schemes allow us to receive either unicast or broadcast replies from the client. Using broadcast instead of unicast eliminates the link-level retransmissions of the requests, which allows us to estimate more accurate the loss rate.

Each SMesh node computes a client link quality metric based on the observed loss of a client's DHCP requests or ARP responses, using the following weighted average decay function:

$$M_{new} = M_{old} * D_f + Current * (1 - D_f) \quad , \quad 0 < D_f < 1$$

where M is the link quality measure and D_f is the decay factor. *Current* is a constant value which is set to 0 if the access point did not receive any DHCP or ARP probe packets responses in the expected time, or is set to a maximum value if a probe packet is received. The access point calculates this function every second for each client in its vicinity. SMesh uses a decay factor of 0.8 to make the protocol resilient to occasional wireless losses of the probe packets, while maintaining its adaptability to network condition changes. SMesh uses a *Current* value of 50 to allow integer calculations with discrete mapping. The tie breaker between two access points having the same integer metric (in the range of 0 to 50) is according to the lowest IP of the access point.

Many wireless devices allow applications to capture packets through a monitoring interface. When the mesh node is also equipped with such an interface (as in the case of our Linksys routers), specific radio measurements from the received packet, as well as the complete 802.11 frame, is available to SMesh, as follows:

1. RSSI (Received Signal Strength Indicator)

   RSSI is a measurement of the radio signal strength. If the wireless interface is configured in monitor mode, an additional header is added by the wireless driver, which contains the RSSI information. One thing we must be aware of is that the RSSI value must be in the same range

[1] When a client is receiving data, it needs to send an acknowledgment at the 802.11 level for every packet it receives, which can also be used to monitor connectivity.

of values for all mesh nodes. If different card manufacturers are used, a conversion might need to be performed (e.g., Cisco Systems cards report a maximum RSSI value of 100, while Atheros cards report a maximum of 60).

2. 802.11 Retransmission Flag

   Every unicast packet transmitted in 802.11 needs to be acknowledge by the recipient. If the packet or the acknowledgment is lost, the sender retransmits the packet, and sets a retransmit flag in the 802.11 header. The maximum number of retransmissions is usually four. In our case, instead of using a broadcast heartbeat to know when packets are lost on the first transmission, we look at this flag to determine if the packet was lost on the first attempt.

The main advantage of using RSSI versus a loss-rate only measurement is that we can start the handoff process to a better access point before there is any loss in the medium. The initial loss in the medium is usually masked by the 802.11 retransmissions, so the client sees this loss as an increase in latency for these packets. However, RSSI alone is not a good indication of the loss rate of a link, so we use it in conjunction with the loss rate, adjusted with the decay function described above, for measuring the quality of the link.

5.2 Intra-domain Mobility Management

5.2.1 Mobile Client Data Group

A mesh node joins the client Data Group so that it can receive and forward data packets for that client, if it believes it has the best connectivity to the client based on link quality metrics it receives from other nodes.

Nodes in a Client Data Group receive data packets that need to be forwarded to the corresponding mobile client. If more than one node is a member of this group, duplicate packets will be sent to the client by each member of the group.

Our protocol must guarantee that, at all times, there is at least one member in the Data Group of each client, such that the client will be served by at least one mesh node. On the other hand, it would be wasteful to allow more than one node in the vicinity of a client to also be in the Data Group most of the time as this creates duplicate packets. Our protocol balances between these two conflicting goals (availability and efficiency).

5.2.2 Mobile Client Control Group

In addition to the previously described Client Data Group, used to forward data packets in SMesh towards the access points serving the client, the access points in the vicinity of a client join a different multicast group specific to that client, called Client Control Group. The Client Control Group is used to coordinate with other mesh nodes in the client's vicinity regarding link quality metrics and regarding which access point will be the best to serve that client. A mesh node joins a client's Control Group when it receives one heartbeat from the client, and leaves the group after not hearing from the client for some time. Both Control and Data group names are derived from client's IP address. For example, for a mobile client with address 10.A.B.C, a SMesh node will join the client's Control Group at 224.A.B.C and, if needed, the client's Data Group at 225.A.B.C. This maps every client to a set of two unique multicast groups[2].

The link quality metric is shared by the access points periodically by posting it on the client's Control Group. Since only the nodes receiving a heartbeat from a client join the client's Control Group, the multicast overhead is localized only in the vicinity of that client and will not propagate beyond that in the network.

[2]Control Groups and Data Groups are implemented as Spines multicast groups.

5.3 Client Handoff

Each mesh node has its own IP address that allows it to communicate with other mesh nodes. However, in order to provide a completely transparent handoff to clients, mesh nodes advertise a virtual gateway IP address to all clients in their DHCP offers and acknowledgments (`DHCPOFFER` and `DHCPACK`). Mobile clients set their default gateway to this virtual IP address regardless of which access point they are connected to. This way, mobile clients get the illusion of being connected to a single access point that follows them as they move. The IP address of the default gateway only appears in the DHCP offer. In all other IP communication with mobile clients, the default gateway does not even appear in the IP packets. It can be set to any valid IP address in the client subnet as the communication with the mobile clients is solely based on MAC addresses.

Our handoff mechanism uses gratuitous ARP messages to instantaneously change the access point used by the mobile client. A gratuitous ARP is an ARP reply packet that is not sent as a reply to an ARP request, but is rather sent in the local network voluntarily. Upon receiving such a packet, a host will update its ARP cache with the value it received. Typically, gratuitous ARPs are used by hosts to advertise their new hardware address when their network card is changed.

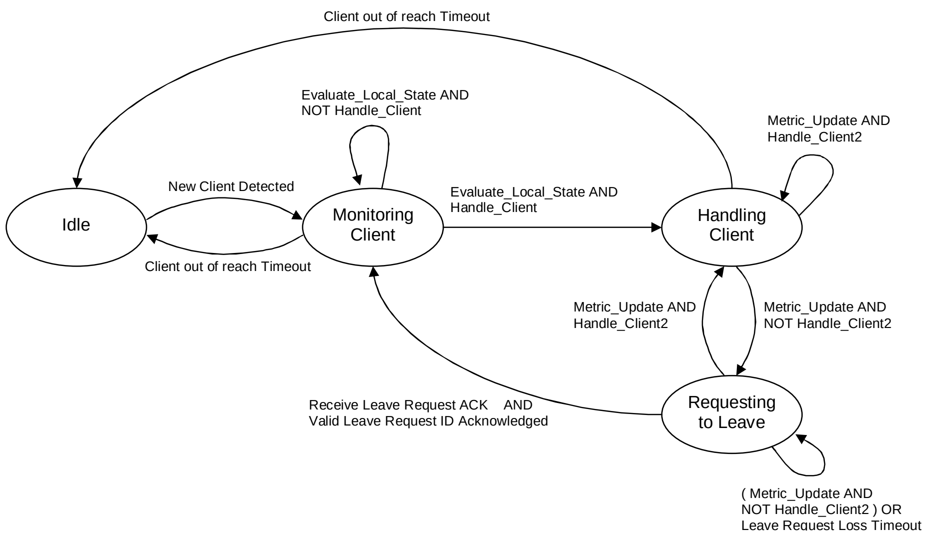
When a SMesh node believes it has the best connectivity with the client and decides to serve that client, it sends a gratuitous ARP as a unicast, directly to the client, thereby changing the MAC address of its default gateway. Subsequent packets sent by the client will be sent to the new access point, following the new hardware address. All operating systems that we have tested accept gratuitous ARPs and begin using the new MAC-IP mapping immediately.

In addition to sending a gratuitous ARP to the mobile client, when a node believes it has the best link quality to a mobile client, it joins its Data Group so that packets destined to the client start flowing through this access point. If another node is also a member of the Data Group, packets destined to this client are forwarded to both mesh nodes, and each of them forwards the packets directly to the mobile client. The mobile client may receive duplicate packets at this time. Using multicast helps achieve uninterrupted connectivity during handoff by: (1) sending packets through multiple access points to the mobile client, to deal with unexpected client movements while the best access point for the client is chosen, and (2) avoiding loss while route changes take place in the wireless mesh.

A mesh node that joins the Data Group of a mobile client immediately sends a metric update on the Control Group to inform any other node of its latest metric, noting that it is now a member of the client's Data Group. When a mesh node that is a member of the Data Group receives a link quality metric update that shows that a different node in the Data Group is better connected, it issues a *Leave Request*. Leave Requests, sent on the Control Group, are piggy-backed on link quality metric updates. A Leave Request can be acknowledged only by a node in the Data Group that believes that it has the best connectivity to the client. A node may leave the Data Group if and only if its request is acknowledged by at least one other node.

The state machine for handling mobile clients is depicted in Figure 3, and the pseudocode depicting our algorithm is shown in Figure 4. Note that a node checks periodically (line A4) if it should service the client, instead of checking immediately after receiving a metric update, to be less aggressive in taking a decision. However, nodes that are already servicing the client check their state immediately after receiving an updated metric (line F2) to service the handoff as fast as possible. During disagreements, more than one node may be a member of the Data Group for some time, until the disagreement is resolved.

When a node issues a *Leave Request*, it includes a unique id that increases each time the mesh node enters the RequestingToLeave state (line B11). A node can acknowledge a *Leave Request* only if it is currently the one handling the client (line D2). Note that a node cannot leave unless it receives an acknowledgment with the ID used in the last *Leave Request* (line E2). This mechanism guarantees that at least one node is a member of the Data Group at all times, unless this node crashes. Our experiments show that this usually lasts less than a quarter of a second during handoffs.


Client out of reach Timeout
Evaluate_Local_State AND NOT Handle_Client
Metric_Update AND Handle_Client2
New Client Detected
Idle
Monitoring Client
Evaluate_Local_State AND Handle_Client
Handling Client
Client out of reach Timeout
Metric_Update AND Handle_Client2
Metric_Update AND NOT Handle_Client2
Receive Leave Request ACK AND Valid Leave Request ID Acknowledged
Requesting to Leave
(Metric_Update AND NOT Handle_Client2) OR Leave Request Loss Timeout

<u>Conditions</u>

Handle_Client:	My_Metric > Highest_Metric(Data Group) * Threshold AND My_Rank(Nodes in MonitoringClient state) <= Maximum_Concurrent_Joins (in our case 2)
Handle_Client2:	My_Rank(Nodes in HandlingClient or RequestingToLeave state) == 1
My_Rank(list):	sort list in decreasing order of their metric value, then by IP address to break ties, and return index of local node

Figure 3: State machine for handling mobile clients.

6 Fast Inter-domain Handoff Protocol

Communication between mobile clients and the Internet is relayed through the closest Internet gateway to improve wireless usage. As mobile clients move within the wireless mesh network, they may get closer, network-wise, to a different Internet gateway in the same island, or they may move to a different wireless island. In this case, the anycast packets, which are forwarded to the closest Internet gateway, will no longer reach the original gateway, and therefore a solution is required to maintain existing connections.

Mobile clients in SMesh work on a private network, and a Network Address Translation (NAT) is required at the Internet gateway when communicating with an external host. Each Internet gateway has a different external IP address. Applications using TCP, and in some cases, applications running on top of UDP require packets to be forwarded through the initial forwarding Internet gateway through the entire life of the connection. Changing one end-point of the connection (the IP address of the Internet gateway) is often impossible without breaking the existing connection, and therefore it is better for the handoff mechanisms to mask this problem inside the mesh network.

One potential solution is to exchange complete connection information (NAT tables) between the Internet gateways periodically and forward packets to the original owner of the connection using the wired connectivity. Such a solution can only be as fast as the time between two periodic NAT table exchanges, and cannot support real-time traffic such as VoIP. To support real-time traffic, one can advertise connection information to all the Internet gateways when the NAT entries are created. However, this technique tends to be wasteful, as not all mobile clients may move and change their Internet gateway. The problem is most notable when clients are browsing the Internet, as many

```
// Abbreviations: DG = data group, CG = control group, LR = leave request

States = {Idle, MonitoringClient, HandlingClient, RequestingToLeave}
LR_ID = 0

A1. New_Client_Detected(client $i$):
A2.    Join(CG$_i$)
A3.    state$_i$ = MonitoringClient
A4.    Periodically(Evaluate_Local_State($i$))
A5.    Periodically(Monitor_Client($i$))
A6.    Periodically(Send_Metric_Update(CG$_i$))

B1. Evaluate_Local_State(client $i$):
B2.    if (state == MonitoringClient)
B3.       My_Rank = Compute_My_Rank(CG$_i$ Members in state == MonitoringClient)
B4.       if (My_Metric$_i$ > (Highest_Metric(DG$_i$ Members) * Threshold) and My_Rank <= 2)
B5.          Join(DG$_i$)
B6.          Send_Gratuitous_ARP($i$)
B7.          state$_i$ = HandlingClient
B8.    else if (state == HandlingClient)
B9.       My_Rank = Compute_My_Rank(DG$_i$ Members)
B10.      if (My_Rank != 1)
B11.         LR_ID$_i$ = LR_ID++
B12.         Send(LR$_{LR\_ID_i}$)
B13.         state$_i$ = RequestingToLeave
B14.   else if (state == RequestingToLeave)
B15.      My_Rank = Compute_My_Rank(DG$_i$ Members)
B16.      if (My_Rank == 1)
B17.         state$_i$ = HandlingClient
B18.   if (current_state$_i$ != previous_state$_i$)
B19.      Send_Metric_Update(CG$_i$)

C1. Compute_My_Rank(list):
C2.    sorted_list = new list sorted in decreasing order of metric value,
              using node_id to break ties
C3.    return the rank/index where local node is located in sorted list

D1. Receive_LR(client $i$):
D2.    if (state$_i$ == HandlingClient)
D3.       Send_ACK(LR$_i$, ID(LR))
D4.       Send_Gratuitous_ARP($i$)

E1. Receive_LR_ACK(client $i$):
E2.    if (state$_i$ == RequestingToLeave and ID(LR_ACK) == LR_ID$_i$)
E3.       Leave(DG$_i$)
E4.       state$_i$ = MonitoringClient

F1. Metric_Update(client $i$):
F2.    if (state == HandlingClient or state == RequestingToLeave)
F3.       Evaluate_Local_State($i$)

G1. Client_out_of_reach_timeout(client $i$):
G2.    if (I_am_member(DG$_i$))
G3.       Leave(DG$_i$)
G4.    Leave(CG$_i$)
G5.    state$_i$ = Idle
```

Figure 4: Pseudocode for deciding when to join and leave the Control and Data Groups.

connections are established for each website and, all of these information, which is relevant only for a small amount of time, would be sent to all of the Internet gateways.

Our inter-domain handoff protocol provides transparent mobility on a NATed network with real-time performance. We treat UDP and TCP connections separately, detect the existing owner (the Internet gateway from which the connection was initiated) of a connection, and forward existing connections through their original owners[3]. Figure 5 shows the flowchart of the protocol, explaining

[3]One can potentially spoof the address of the original owner to reduce the routing overhead of our protocol. However, egress filtering is commonly used at network routers and will prevent spoofed packets from leaving their network.

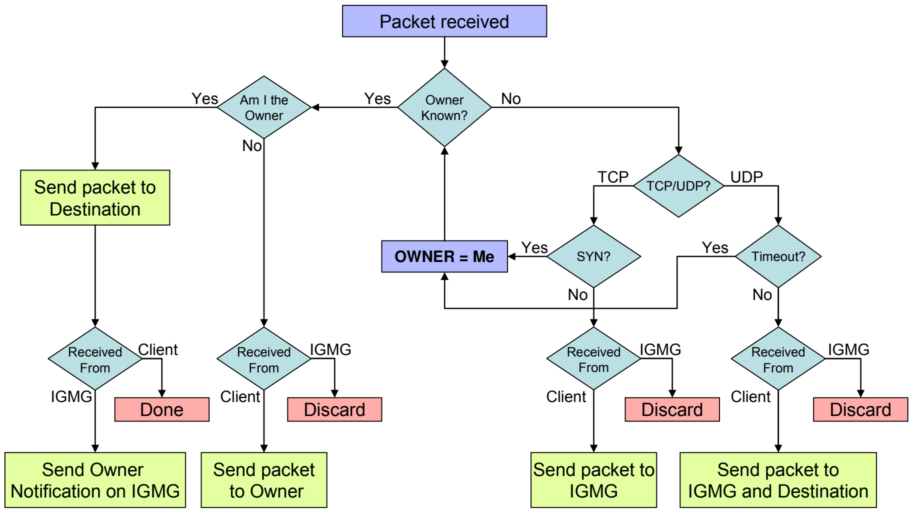


Figure 5: Inter-domain handoff flowchart.

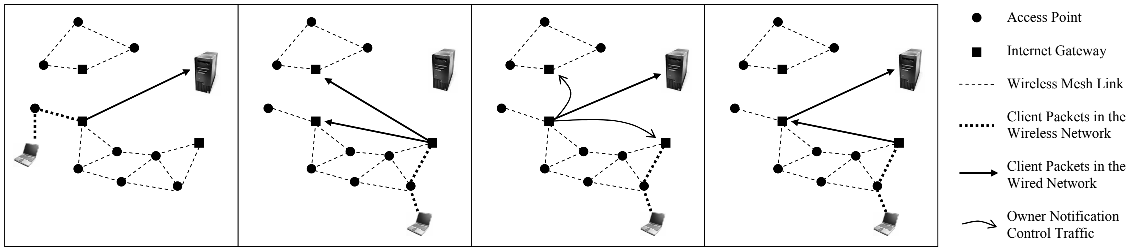

Access Point
Internet Gateway
Wireless Mesh Link
Client Packets in the Wireless Network
Client Packets in the Wired Network
Owner Notification Control Traffic

Figure 6: TCP forward handoff: (a) Connection establishment (b) Handoff Phase 1 (c) Handoff Phase 2 (d) Handoff completed

how a packet is handled at an Internet gateway.

6.1 TCP Connection Handoff

A TCP session requires that source and destination IP addresses and ports remain constant during the life of the connection. Our mobile clients run in a NAT address space, and although connections are end-to-end, the Internet destination regards the source address as that of the Internet gateway that sent the first SYN packet. When a mobile client moves closer to a different Internet gateway, the new gateway must forward all packets of each existing connection to the original gateway that initiated that connection. On the other hand, new connections should use the Internet gateway that is closer to the client at the current time, and not be forwarded to an old gateway.

In TCP, a SYN packet indicates the creation of a connection and generates a NAT entry, while a FIN packet indicates the destruction of the connection. If an Internet gateway receives a TCP packet that is not a SYN and it does not have an entry for that connection in its NAT table, it forwards that packet to the IGMG group (Internet Gateways Multicast Group). The original owner of the connection (the one that has it in its NAT table) relays the packet to the destination, and sends a message to the IGMG group, indicating that it is the connection owner for that NAT entry. Then, any gateway that is not the connection owner, will forward packets of that connection to the respective owner, finalizing the connection handoff process. Figure 6 shows the stages of such a TCP connection handoff.

If packets arrive at an Internet gateway at a fast rate, several packets may be sent to the IGMG group before the connection owner can respond. If no Internet gateway claims the connection within a certain timeout (in our implementation 3 seconds), the new gateway claims the connection, forwarding the packets directly to the Internet destination. This will break the TCP connection, which is the desired behavior in such a case, since it is likely that the original owner crashed or got disconnected. Causing the Internet host to close the connection avoids connection hanging for a long period of time (TCP default is 2 hours).

6.2 UDP Connection Handoff

Most real-time applications use the best effort UDP service and build their own protocol on top of UDP to meet specific packet latency requirements. Some applications, such as DNS, do not establish connections between participants. Others, such as SIP in VoIP, establish specific connections defined by a pair of an IP address and a port at both ends of the connection.

When an Internet gateway receives a UDP packet with a new pair of source and destination addresses or ports, it cannot distinguish between the case where this is the first packet of a new connection, and the case where the packet belongs to an existing connection established through a different Internet gateway.

We classify UDP traffic on a port number basis as *connection-less* and *connection-oriented*, and choose connection-oriented as the default protocol. Connection-less UDP traffic is forwarded directly after receiving it from the mesh network, on the current shortest path. DNS and NTP traffic falls into this category.

Upon receiving a new connection-oriented UDP packet that has an Internet destination, an Internet gateway relays that packet to its destination, and also forwards it to the multicast group that all Internet gateways join (as opposed to the TCP case, where the access point only sends packets to the multicast group). If the UDP packet belongs to a connection that was already established, the Internet gateway that is the original owner of the connection also relays the packet to the destination, and sends a response to the IGMG group. After receiving the response, the initial gateway will forward subsequent packets directly to the original gateway, and will no longer relay UDP packets of that connection (with the same source and destination addresses and ports) to the Internet. If a response does not arrive within a certain timeout (in our implementation 500 milliseconds), the Internet gateway will claim ownership of the UDP connection, will stop forwarding packets of that connection to the IGMG group, and will continue to relay packets to the Internet.

6.3 Discussion

Due to handoff and/or metric fluctuations, there is a possibility that packets coming from a mobile client and belonging to the same flow alternate between two Internet gateways. This may lead to more than one gateways claiming the ownership of the connection. We encounter such case in TCP when a client retransmits a SYN connection request, and this request is routed through a different Internet gateway. In UDP, such case may occur when two different Internet gateways start forwarding client packets for the same connection at about the same time. One solution for TCP is to delay ownership decision until a full three-way TCP handshake is seen by the Internet gateway. For UDP, when there is more than one ownership request in parallel, the gateways can decide the rightful owner of the connection based on feedback traffic from the end-host or lowest IP address.

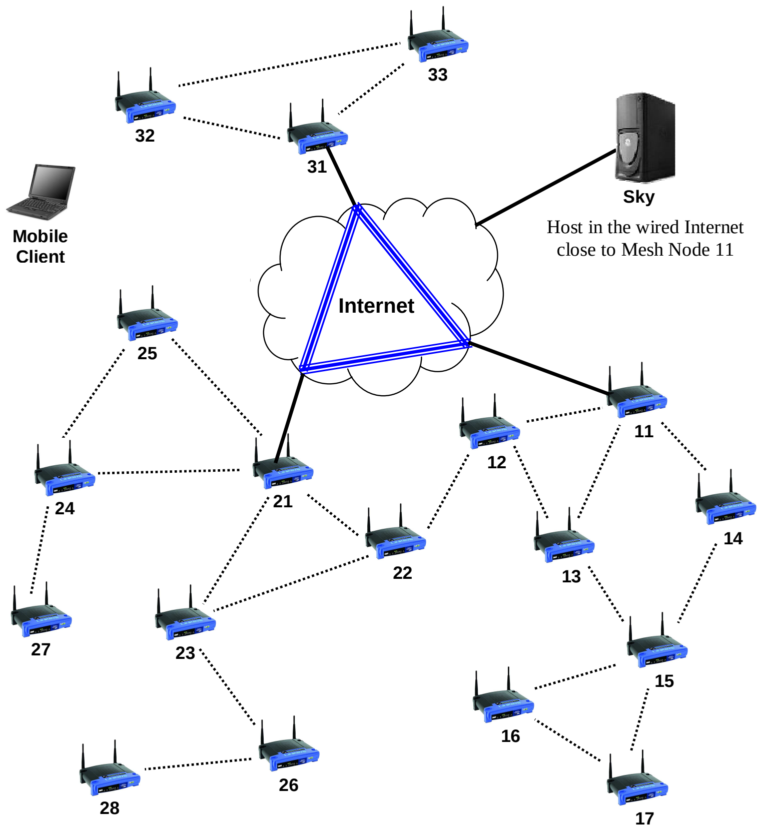


Figure 7: The SMesh multi-homed wireless mesh testbed.

7 Experimental Results

7.1 Setup

We deployed our system on 18 Linksys WRT54G wireless routers across several floors in four buildings at Johns Hopkins University. Each of the routers is equipped with one radio configured in ad-hoc mode. Transmit power of the access points was set to 50 mW, and the data-rate to 11 Mbps. In each Linksys router, we replaced the original firmware with OpenWrt [34], an open-source Linux environment suitable for running the SMesh software.

We used two laptop computers, each equipped with a Lucent Orinoco 802.11b Mini-PCI card in ad-hoc mode as mobile clients. We used Linux for all experiments that required precise timing measurements. Windows XP was used for a TCP throughput experiment, also showing how SMesh operates with different client platforms. No software other than the benchmarking programs was installed on the laptop computers.

The topology of the wireless testbed used in our experiments is shown in Figure 7. The topology consists of one main island with two Internet gateways, and another smaller island with one Internet gateway. The islands are disconnected due to a large open grass area between the buildings. However, a mobile client located between the two islands can reach both networks. Each of the Internet gateways is part of a different domain on the campus network and within 6 hops of each other through the wired network. Unless otherwise specified, the topology between the access points was static during the experiments. Each access point box has an identifier, referred to as node ID. The node ID of Internet gateways ends with digit 1 (mesh nodes 11, 21, and 31). The closest Internet gateway of mesh nodes is given by the prefix of the access point node ID (i.e., node 23 uses node 21 as its Internet gateway). In addition, the node IDs are ordered by number of hops from the gateway (i.e., node 23 is equal or less number of hops from its gateway than node 24).

The SMesh DHCP server was set to issue lease times to clients for 90 seconds. The SMesh monitor was set to unicast ARP requests to the client and to use loss rate and RSSI when computing the client metric. For the link quality measure we used a *Current* value of 50, and we set the decaying factor, D_f, to 0.8. The *Threshold* for joining the Client Data Group was set to 12%. In our experiments these numbers provided the best trade-off between the granularity of the metric and handoff responsiveness.

Our experiments were performed with one mobile client inside SMesh communicating with a Linux machine that resided in the wired network (Internet), one wired hop away from the mesh Internet gateway. The SMesh client will be referred to as *Client* and the Linux box from the Internet as *Sky*. In the experiments we sent full-duplex VoIP traffic, one stream from Client to Sky and another from Sky to Client. The VoIP traffic consisted of 160 byte UDP packets sent every 20 ms at a rate of 64 Kbps. This traffic is equivalent to that of G.711, the standard encoder used for VoIP communication. We focus our experiments on VoIP as a representative application that poses severe latency requirements.

We first performed a stationary test to set the baseline of our experiments with a moving client. We then proceeded to move across several buildings, starting and/or ending at the same location as the stationary experiment. We then show how TCP behaves as we move across the mesh. We tested the fail-over performance of our protocol when the access point of the *Client* or its closest Internet gateway suddenly crashes (we disconnected the power of the Linksys router). Finally, we added emulated clients into the system, and determined how the management overhead of the mesh network increases as the number of clients in the system grows.

For each test we monitored the one-way latency of each packet, the number of lost packets, and the number of duplicate packets. The one-way latency was adjusted taking into account the difference between the clocks at the *Client* and *Sky* machines. For VoIP communication it was also important to track the delay jitter as well as how many packets arrived within 100 ms, the rest being considered lost by the audio codec. Based on tcpdump logs we reconstructed the handoff decisions and computed the communication overhead. We show the handoff information in the graphs, noting also the number of wireless hops from each mesh node to the Internet gateway. Note that the Client is connected to the access point through a wireless link, and therefore its latency is influenced by this additional link. When we state the number of hops of an access point we do not count the wireless hop from the client to its current access point.

7.2 Measurements

7.2.1 Stationary client

This test was performed with the mobile client being stationary, in a fixed position for the duration of the entire test. UDP traffic consisting of 15,000 packets (about 5 minutes) was sent simultaneously in each direction: from the Internet box (*Sky*) to a mobile client (*Client*), and from *Client* towards *Sky*. The packet latencies are shown in Figure 8 and Figure 9. The dotted line tracks which mesh node is the current access point of the *Client*. Vertical lines represent the moments when a gratuitous ARP that caused a handoff was received. For example 12_1 on the right side of the graph refers to node 12 in our topology, which is 1 hop away from the Internet gateway. We notice that even though the client was stationary, its access point changed between two nodes in its vicinity: box 12_1, and then 13_1. This happens because the wireless connectivity varies, and over time, different access points have a better connection to the *Client*.

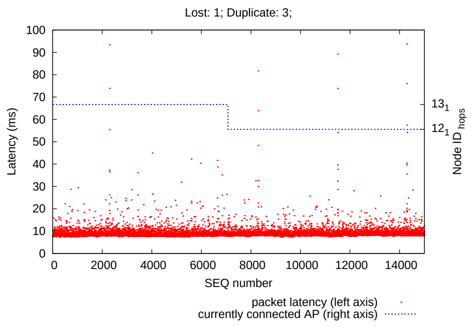


Figure 8: Stationary client. Latency of the packets received by *Client*.

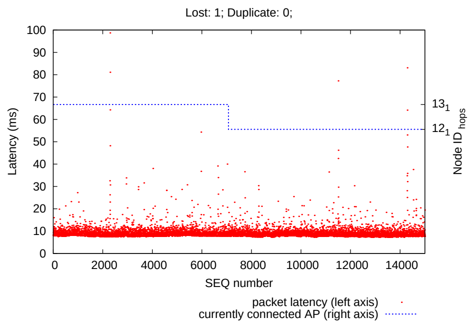


Figure 9: Stationary client. Latency of the packets received by *Sky*.

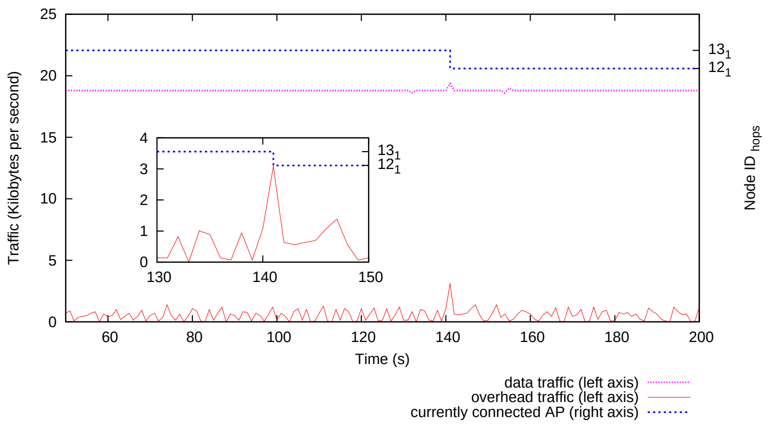


Figure 10: Data and SMesh overhead traffic for stationary client experiment. The subgraph shows a zoomed view of the overhead during handoff.

For the first stream (*Client* is the receiver, Figure 8), the number of lost packets was 1 (packet #6,661), and the number of duplicate packets was 3. This amounts to an overhead due to duplicates during handoffs of .01%. During this experiment, 4 packets (0.02% of the total traffic) were delayed by more than 100 ms, and all packets arrived in less than 200 ms. The handoff process started at packet 7,076, so no loss occurred during the handoff. As expected, the duplicate traffic occurred only during the handoffs[4].

The reverse stream (*Sky* is the receiver, Figure 9) had also 1 packet loss (packet #11,508), and no duplicate packets. Only 1 packet arrived later than 100 ms, but before 200 ms. In all the tests when the Internet box (*Sky*) is the receiver, the number of duplicate packets must be zero: the packets are sent only once by the client (only to its current access point), in contrast to the other direction (from Sky to the Client).

The number of duplicate packets are about one order of magnitude less than the results provided in our original paper [6]. One reason for the improvement is that originally we were re-evaluating a node's status at a periodic timeout, while in our current version a node on the Data group evaluates whenever it gets a metric update, and in addition it sends out its metric immediately if it contains a *Leave Request*[5] (Figure 4 line F3).

Figure 10 shows the overhead of our system in comparison with the data traffic. The data traffic represents the data traffic sent and received by the client during the experiment. The overhead traffic represents the data traffic sent and received by one of the mesh nodes in the client vicinity (mesh node 13). The bandwidth measured is higher than the full duplex 64 Kbps UDP stream we sent, due to the IP and UDP headers that accumulate on the relatively small (160 byte) packets.

Control traffic from our system is represented as the bottom traffic line. It combines the traffic from Spines (joins and leaves from multicast groups, hello keep-alive messages, link state updates) and the traffic from client's Control Group (link quality updates, leave requests and acknowledgments). Spines sends keep-alive messages of 40 bytes every 4 seconds. Link state updates are sent only when the mesh topology (formed by access points) changes. Join and leave messages are sent only when a SMesh node (access point) joins or leaves a group. These types of messages are aggregated such that a single Ethernet packet can contain up to 90 updates. In order to keep track of the clients (posting link quality measures, sending ARP packets), a SMesh node sends about 20 bytes in each update, sent every few seconds, for each client in its vicinity.

As we can see in Figure 10, a handoff takes place around second 140. The overhead during handoff is shown in detail in the zoomed graph on the left of the figure. The increase in control traffic show the moment when node 12 decided to join the Data Group, and sent a join message to Spines (join and leave operations will generate a state update in the Spines overlay network). As a consequence, there is a small spike in the data traffic since data packets are duplicated. Right after, the old access point decided to leave the client Data Group (it sends a Leave Request and it immediately receives the acknowledgment). All of these happens in less than a second, so all of the overhead related to the handoff is represented by the spike in the control traffic during handoff.

We use the above stationary client results as a baseline for the following tests, to provide an idea of our wireless environment, and to overview the handoff process before a more elaborate scenario.

7.2.2 Intra-domain handoff for a moving client

In this test we move the client from the stationary position of the previous experiment, taking it on a 5 minutes trip across two floors and ending in the original position. We disconnected nodes 21 and 31 from the Internet, leaving only a single Internet connected node in mesh, to demonstrate our intra-domain handoff performance. We used the stairs to move between the floors. During the

[4]We define a "handoff" as the entire interval when duplicate packets are received by the client. Note that a client switches from one access point to another in much less time – the amount of time it takes for a gratuitous ARP packet to arrive from the access point to the client.

[5]As opposed to initial approach when we only send the acknowledgment immediately.

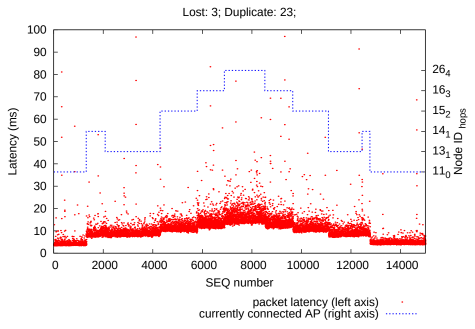


Figure 11: Moving client. Latency of the packets received by *Client*.

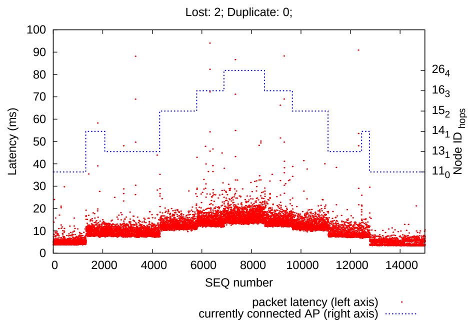


Figure 12: Moving client. Latency of the packets received by *Sky*.

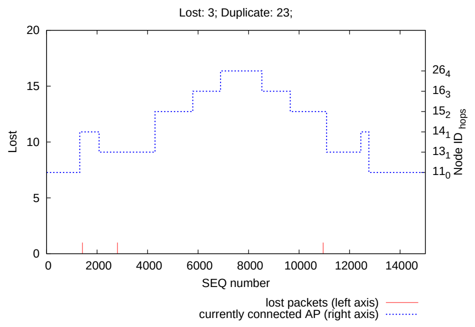


Figure 13: Moving client. Packets lost by *Client*.

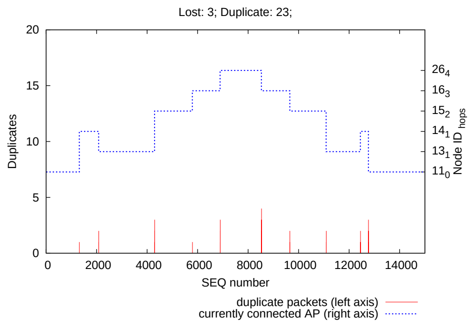


Figure 14: Moving client. Duplicate packets received by *Client*.

test, the client changed its access point 10 times, spanning from zero-hops away (11_0) to four-hops away (26_4) (node 26 needs to reach node 11 when routing traffic to the Internet). Note that the wireless hop between the client and its current access point is not counted in the number of hops in the network (so there is effectively one more wireless hop end-to-end).

The latency graphs for each of the two VoIP streams are shown in Figures 11 and 12 respectively. Each additional hop on the path from the Client to the Internet gateway resulted in an increase in packet latency: between sequence numbers 0 and 1,315 we were zero hops away, between 1,315 and 4,298 one hop away, and between 4,292 and 5,794 two hops away. The number of packets that did not arrive within 100 ms on the Client and Sky was 25 and 13, respectively. All packets arrived within 200 ms.

The data stream towards the *Client* had 3 packets lost, and 23 duplicate packets. Figure 13 presents the cumulative number of lost packets in a window of last 20 packets. The first loss occurred at packet 1,419, about 100 packets after the handoff; this loss happened due due to loss in the medium. The second loss is far from any handoff, and happened due to loss in the medium as well. The third loss, however, happened at packet 10,952, and the handoff started at packet 11,094, or about 2.8 seconds after the loss. This loss contributed to lowering the metric, and to triggering the handoff. There is a possibility that the loss could have been prevented if the handoff would have happened earlier. While possible, our threshold (set at 12%) attempts to balance stability with handoff performance, and new information about a sudden drop in signal quality from a node takes time to propagate to other nodes. However, most of the handoffs were performed in a timely fashion without any loss before, during, or after the handoff. None of the losses in the experiment happened during handoff itself.

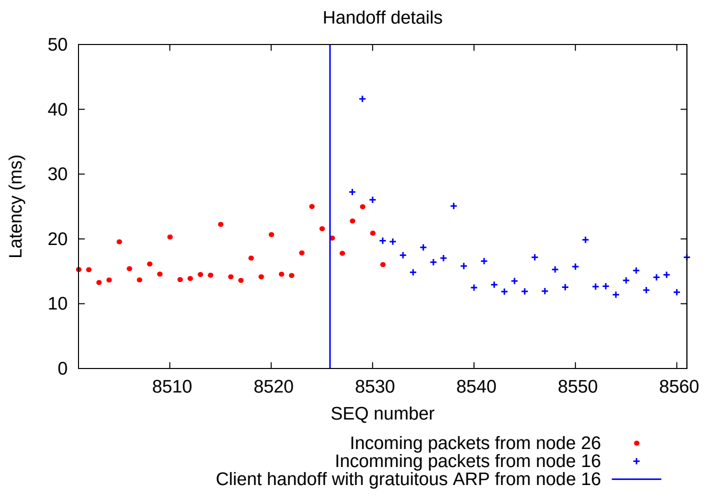


Figure 15: Moving client. Zoomed view of the packets received by *Client*.

Figure 14 shows the cumulative number of duplicate packets received in a window of last 20 packets. Note that duplicate packets happen only during handoffs. We can see that there is a correlation between the distance between the mesh nodes involved in the handoff and the number of duplicate packets. For example, nodes 15 and 16 are direct neighbors, and one to two duplicates were seen during a handoff between these nodes. In contrast, nodes 16 and 26 are furthest from each other, 6 wireless hops total, and three to four duplicates were recorded. In our approach, a node needs to learn about someone taking over the connection, request to leave, and receive an acknowledgment, before it can leave the Data Group associated with the *Client*. In addition, the multicast leave operation needs to propagate through the network.

The number of duplicates in our experiments show a lower bound for our network; one can allow for more time to elapse before acknowledging a leave request to ensure that the state is fully propagated through the network before a multicast leave operation is issued. This will usually be a function of the diameter of the network and the timeouts for propagating state updates in each hop. While we did not experience any loss during handoff, allowing for longer period of time may be useful in other deployments.

The stream towards *Sky*, depicted in Figure 12, had 2 lost packets and, as expected, 0 duplicates.

Figure 15 represents a zoomed view of the handoff happening at sequence 8,526, for the same experiment. The dots represent the packets forwarded by the previous access point (node 26), and the crosses represent the packets forwarded by the new access point (node 16). The vertical line shows when the client received a gratuitous ARP from the new access point (node 16). This is the handoff that experienced the most number of duplicates, and at the same time between the nodes that are the most number of hops away from the Internet gateway and from each other.

The gratuitous ARP from node 16 was received just before packet 8,526. However, there are no duplicates until packet 8,528. When a node makes a local decision to start handling the client, it issues a gratuitous ARP in addition to a multicast join for the Data Group for the client. However, this multicast join needs to propagate and routes need to be established before packets start flowing towards the new access point. It took about 25 ms for this to happen, which is consistent with the number of hops between the nodes and our choice of timers[6] in the system. When communicating with a node in the Internet, this delay depends on the number of hops from the node joining the

[6]In Spines, each overlay node waits 5 ms before forwarding the update to its neighbors. This allows the overlay to potentially aggregate more updates and scale.

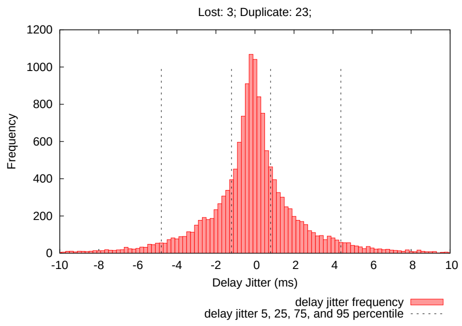


Figure 16: Moving client. Delay jitter of the packets received by *Client*.

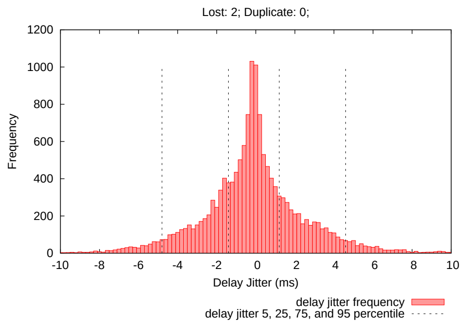


Figure 17: Moving client. Delay jitter of the packets received by *Sky*.

client Data Group and the Internet gateway. Starting at packet 8,528, there are four duplicate packets received by the Client. As previously explained, the number of duplicate packets depend on the number of hops between the nodes involved in the handoff. We can see a slight increase in latency during handoff, which is caused by the increase in congestion in the wireless network. The latency drops slightly afterwards as the new access point is one hop closer to the Internet gateway.

Figure 16 and 17 show the delay jitter, or Packet Delay Variation [22], of the VoIP stream towards the *Client* and towards *Sky*, respectively. A big variation can have a negative effect on the play-out buffer at the end-points of the VoIP stream. In our test, the Inter Quartile Range (IQR), which represents the difference between the 25 and the 75 percentile, was just 2.6 ms. The difference between the 5 and the 90 percentile was under 10 ms, which is considered excellent for VoIP communication. One can also see in Figure 15 that there is a light increase in jitter during handoff. Considering that a jitter of less than 20 ms is considered excellent by VoIP applications, the quality of the voice is not impacted by the jitter experienced in the mesh network.

7.2.3 Inter-domain handoff for connection-oriented UDP streams

In this test, we walked with a mobile client from the 3^{rd} floor of a building located in the main island to a hallway in the second floor, followed by going down to the ground floor. Then, while walking outside on an open grass area we end up reaching the second island. This movement results in a few access point handoffs and at least three Internet gateway handoffs.

Figures 18 and 19 show the one-way packet latency for packets received at *Client* and *Sky*, respectively. The horizontal lines marked IG_{HO} separate the graph into three areas defined by the Internet gateway forwarding the mobile client's packets to and from the Internet. An inter-domain handoff happens when the dotted line, showing the current access point serving the client, crosses one of the horizontal line.

The initial latency of just about 5 ms represents the latency when going through the Internet gateway that is the owner of the connection. We then move between three different access points, each one hope from the original gateway, and the latency stays constant at around 9 ms. The following handoff, around packet 5,000, shows the first inter-domain handoff in the system; the new node handling the client, node 22, is closer to a different Internet gateway, node 21. Although the number of wireless hops stayed the same, the latency increases as there is additional processing at the Internet gateways and the wired network needs to be crossed. However, the increase is latency is not symmetrical. The reason is that there is additional overhead in processing packets that flow towards the Internet as they need to be sent to our SMesh process an additional time.

Figure 20 shows the packets lost at *Sky*. There were only 8 packets lost, but no losses during the inter-domain handoffs. The number of packets that arrived after more than 100 ms was 2 in the

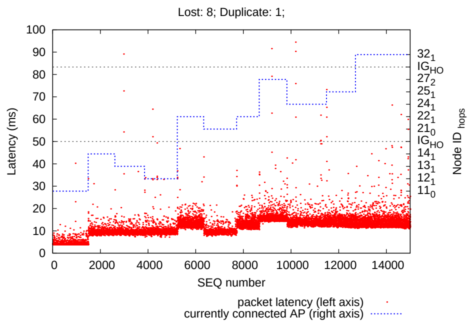


Figure 18: Inter-domain test. Latency of the packets received by *Sky*.

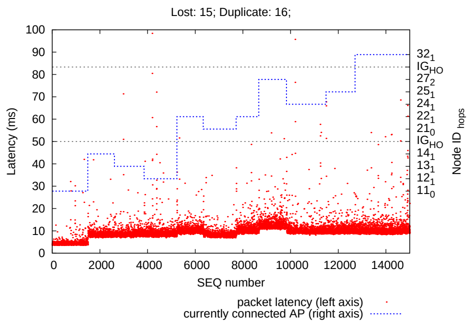


Figure 19: Inter-domain test. Latency of the packets received by *Client*.

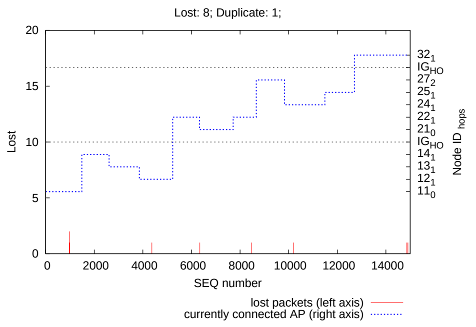


Figure 20: Inter-domain test. Packets lost by *Sky*.

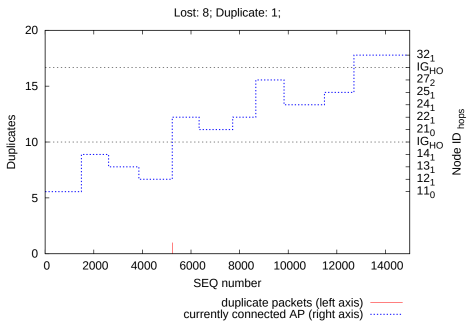


Figure 21: Inter-domain test. Duplicate packets received by *Sky*.

stream from *Sky* to *Client* and 0 in the stream from *Client* to *Sky*. All packets were received within 200 ms. Considering the total number of packets (15,000 in each direction), very few packets were lost or delayed.

In Figure 21 we show the duplicate packets received by *Sky*. These duplicates are caused by inter-domain handoffs. There was only 1 duplicate packet on the stream in the entire experiment, and it occurred during the first Internet gateway handoff. Since Box 21 was not aware initially whether the packet belong to a new or an already existing connection, it sent the traffic both to the IGMG group and to the final destination (as explained in Section 6.2). Because node 11 already had a connection established for that stream in its NAT entries, it forwarded the packet to the Internet destination, and at the same time, it notified the other gateways that it is the owner of the connection, by sending an acknowledgment to the IGMG group. As soon as node 21 received an ownership acknowledgment from node 11, it stopped relaying packets to *Sky* and started forwarding the packets to node 11. Since there was only 1 duplicate packet received by Sky, the inter-domain handoff took less than 20 ms to complete. Note that after the notification, all gateways learned about the ownership of that connection. This is the reason there are no duplicates in the second gateway handoff, from node 21 to node 31 that occurs before packet 14,000.

7.2.4 Peer-to-peer UDP test

During this experiment one mobile client is stationary while the other walks through the same path as the previous experiment. Routing decisions are based on the path that decreases the number of wireless hops between the clients in the hybrid wired-wireless overlay network. The stationary

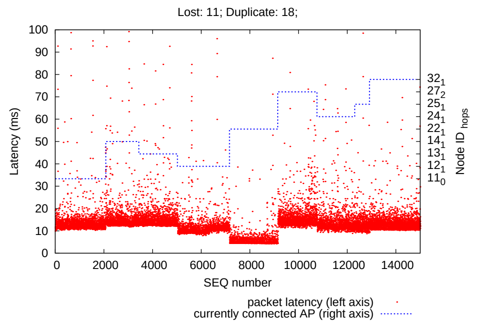


Figure 22: P2P test. Latency of packets received by *Moving Client*.

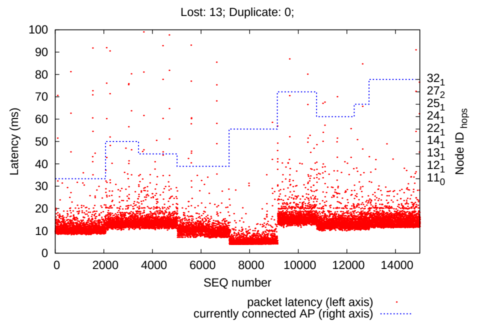


Figure 23: P2P test. Latency of packets received by *Static Client*.

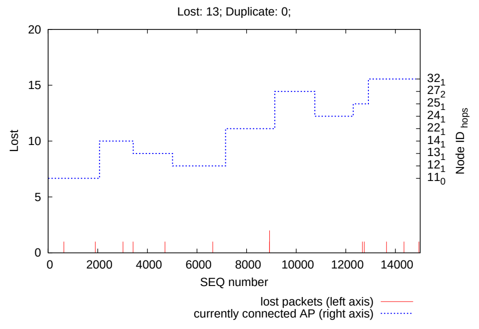


Figure 24: P2P test. Packets lost by *Static Client*.

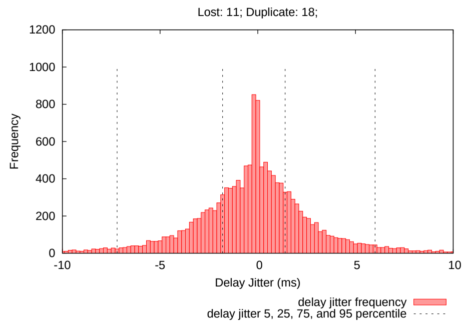


Figure 25: P2P test. Delay Jitter of the packets received by *Mobile Client*.

Client is connected to node 22 at all times; the Client does not experience any handoff throughout the experiment. Figures 22 - 25 present the results of this experiment.

Figures 22 and 23 show the one-way latency of packets as they are received at each client. The initial latency represents 3 wireless hops plus 1 wired hop. This is because there is one wireless hop between the mobile client and node 11, plus one wired hop between node 11 and node 21, plus two wireless hops between node 21 and the stationary client who is connected to node 22. Note that, network wise, this corresponds to one wireless hop. A direct route that did not use the hybrid wired-wireless route would have used an additional wireless hop in order to route packets between the clients.

Around packet 2,000, the latency increases slightly as mesh nodes 13 and 14 require one additional wireless hop through the hybrid route towards the stationary client. Around packet 5,000, the client connects to mesh node 12. Instead of using the wired-wireless hybrid path, the node uses a direct path because the cost in terms of wireless transmissions is the same. However, there is a decrease in latency of about 3 ms, which represents the cost of going through the wired network plus one additional application level router. The two clients connect through the same access point around packet 7,000. The mesh node then connects through a node that is two direct wireless hops away, and then one wireless hop away, until packet 13,000. Then, the mobile client moves to a node that resides in a different island, and must use the hybrid path to reach the stationary client. Note that the latency is similar to the one at the beginning of the experiment, where a different overlay link through the wired network was used to forward packets to the stationary client.

Overall, 13 packets were lost in one direction and 11 in the other. Figure 24 shows the lost packets at the stationary client, who experienced the most number of losses. Loss is represented as

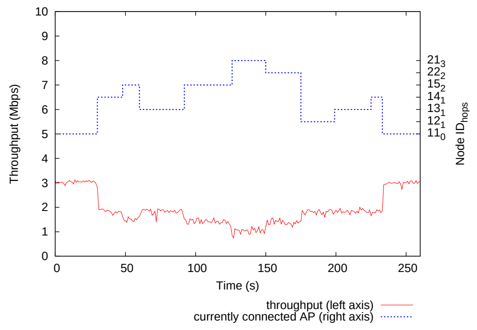


Figure 26: TCP throughput test with one Internet gateway.

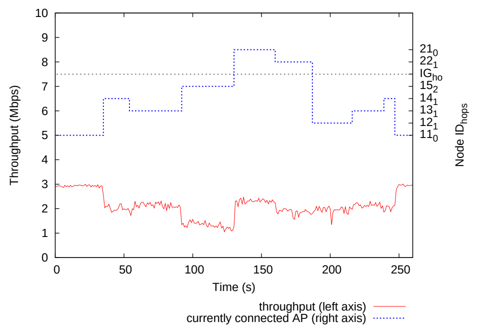


Figure 27: TCP throughput test with 3 Internet gateways.

cumulative number of losses over the last 20 packets. A maximum of two consecutive packets was lost around packet 9,000. As the wireless medium is shared, a sudden loss may be triggered by a number of factors including external wireless communication or interference from our own wireless network. Also, losses can help to trigger a handoff when in conjunction with the RSSI, the metric of a mesh node that is starting to have better connectivity goes above the threshold. In most real time applications, the effect of a relatively small number of packets being lost can be compensated with no interruption in service or significant quality degradation.

Figure 25 present the delay jitter for the stream received at the mobile client. The Inter Quartile Range (IQR) was just 3.2 ms. This is slightly higher than the IQR experienced during the intra-domain handoff in the intra-domain experiment, but considering that now both the sender and the receiver are in the wireless network, the increase in the IQR is not significant.

There were only 18 duplicates over the 8 handoffs experienced by the mobile client, 51 packets arrived after 100 ms out of which 2 packets arrived after 200 ms. The other client did not experienced any handoff, and therefore there were no duplicate packets in this direction.

7.2.5 Inter-domain TCP handoff

This experiment shows the maximum TCP throughput of the mobile client as it moves throughout two floors, utilizing two different stairs in opposite sides of the building.

Figure 26 shows the test performed with a single Internet connected node (mesh node 11) in the wireless mesh. There were a total of 10 handoffs during this experiment. The throughput is 3 Mbps when the client is connected directly to the Internet gateway. As the number of hops increases, the throughput goes down to about 1 Mbps. The throughput returned back to the original amount when we reached the location where the test was started. TCP connection remained open at all times, and packets kept flowing regularly.

Figure 27 shows the TCP download throughput experienced by the mobile client with all Internet gateways connected to the Internet in our multi-homed wireless mesh network. There were 9 handoffs during this experiment. The throughput is initially the same as in the previous test, at approximately 3 Mbps. As the number of hops increases, the throughput goes down to about 1 Mbps. The inter-domain handoff takes place around second 130, where we see the throughput going up to about 2.3 Mbps. In contrast, in Figure 26, the throughput stays at around 1 Mbps when connected to the same node. The throughput returned back to the original amount when we reached the original location where we started the test. Again, TCP connection remained open at all times, and packets kept flowing regularly.

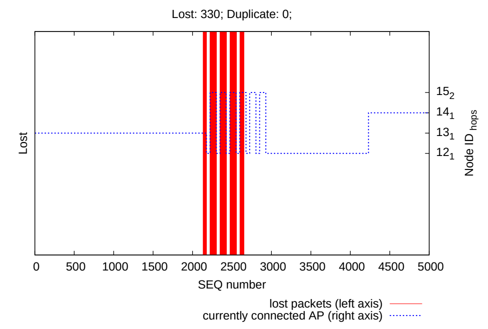


Figure 28: Mesh node and topology fail-over test. Packets lost by *Sky*.

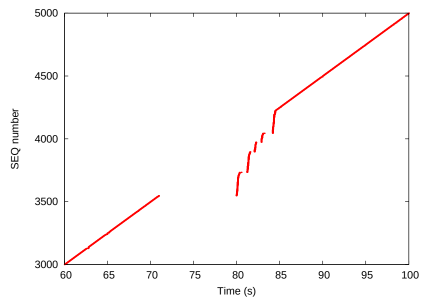


Figure 29: Internet gateway fail-over test in multi-homed wireless mesh. TCP packets received by *Sky*.

7.2.6 Mesh node fail-over

In this experiment we evaluated the fail-over performance of our system when the access point currently serving the client suddenly crashes. We used a stationary client connected to access point 13, sending a VoIP stream to the *Sky* box. As the *Client* was sending packets, we suddenly disconnected the power at node 13. Figure 28 shows the packets lost at *Sky* from the *Client* when node 13 fails. We can see that there are 5 intervals of loss close to each other. The first loss interval occurs as the *Client* keeps sending packets to node 13 after it fails. Shortly thereafter, node 12 notices it does not receive link quality measures from node 13, and sends a gratuitous ARP to the client, forcing its handoff. In our topology, the minimum hop distance routing selects the route between nodes 12 and 15 to go through node 13. After node 13 crashed, node 12 and node 15 do not receive link quality updates from each other, until routing in Spines is repaired. Therefore, both nodes 12 and 15 believe they have the best link quality to the client. They both insist on taking over the connection from the client, sending ARP messages to it. We can see this behavior in the six handoff oscillations depicted in Figure 28. Since node 15 does not have a route to the Internet gateway until the routing protocol in Spines detects the failure (its original route went through node 13), whenever it takes over the *Client*, the data packets are lost. This explains the following intervals of loss after the initial handoff. After Spines detects the failure and the network routes are fixed, packets from the *Client* are no longer lost (both nodes 12 and 15 can reach the Internet gateway). However, it takes a few more seconds for nodes 12 and 15 to send their link quality measures to each other and decide which one should serve the *Client*. Indeed, Figure 28 shows three more handoffs between nodes 12 and 15 until 12 is selected to serve the client.

7.2.7 Internet gateway fail-over

It is interesting to see what happens when the Internet gateway used by a TCP connection suddenly fails. If that Internet gateway is the owner of the connection, then we expect that the connection will break. However, if the Internet gateway is not the original owner of the connection, but rather the one closer to the mobile client that forwards packets to the owner Internet gateway, we expect the mesh network to discover the failure and adjust the routing such that the data packets will reach the owner gateway.

In this experiment we started a TCP connection between *Client* and *Sky* and then moved the client in the vicinity of a different Internet gateway, forcing a gateway handoff to occur. Then we unplugged the power of the current Internet gateway. Figure 29 presents the evolution of a TCP flow where the X axis shows the time and the Y axis shows the packet sequence number. The graph starts after the first handoff from the original gateway. The graph shows about 8 seconds of disconnection required for the mesh network to detect the failure and adjust its routing. After that, it takes a few more seconds for TCP to catch up with the original rate. The network reacting to the failure in a timely manner prevented the disconnection of the TCP connection, overcoming the current Internet gateway crash.

7.2.8 Overhead

In this experiment we measured the management overhead of the system as additional mobile clients are introduced into the network. We evaluated the control traffic required to propagate routing and group membership information, to handle client mobility, and to maintain network's topology. We focus on the overhead of the *control* traffic, as the overhead caused by duplication of data packets was discussed in the previous experiments.

There are five main components of the control traffic:

- Hello messages: The mesh nodes send beacon messages of 40 bytes every 5 seconds in order to discover changes in the topology (node crashes or additional nodes in the system). This traffic does not depend on the number of mobile clients in the system, nor on their mobility.
- Link state updates: These messages propagate information about topology changes. The state update messages are small (under 36 bytes), and multiple states are aggregated in a single packet whenever possible. Since the mesh nodes are stationary and the topology is relatively stable, and because we use reliable state updates, this overhead is negligible (basically 0 in our experiments). Therefore, we do not consider it in our analysis.
- Group state updates: These are the messages used to exchange group membership information between the nodes. The state update messages are also small (under 36 bytes), and multiple states are aggregated in a single packet whenever possible. The number of group state updates is highly related to the mobility and the number of clients. As a client moves, some mesh nodes will join its Control Group and Data Group, while others will leave.
- Gratuitous ARP messages: Gratuitous ARP messages are sent by the members of Data Groups as described in Section 5.3. The size of an ARP packet is 28 bytes. As mobile clients change their access points as they move, the ARP traffic depends mostly on the number of the clients and their mobility.
- Monitoring messages: These are ARP heartbeat packets that are sent (and received) by the access points to assess the quality of the link with a client. In our experiments an access point probes a client every second. This component of the control traffic increases linearly with the number of clients.
- Link Quality updates: Nodes in the vicinity of a mobile client send 68 byte messages periodically, to share information about the link quality between the members of the Control Group, and during handoff. The Link Quality traffic depends on the number of clients and their mobility.

In addition, Internet gateways generate some overhead traffic on the wired network during the inter-domain handoff. Data packets are multicasted over the wired network to all other Internet gateways until the owner of the connection responds. In our tests, this process took between 10 ms and 60 ms. Note that data packets are forwarded in parallel to the end-host and their latency is much less. After the first handoff of a connection takes place, all Internet gateways are informed about the owner of that connection, and therefore new data packets are sent directly to the connection owner. As opposed to the intra-domain overhead, which is only dependent on the number of clients, the inter-domain overhead is directly proportional to the number of connections each client has. However, the traffic generated by the inter-domain handoff is small, and uses only the wired network.

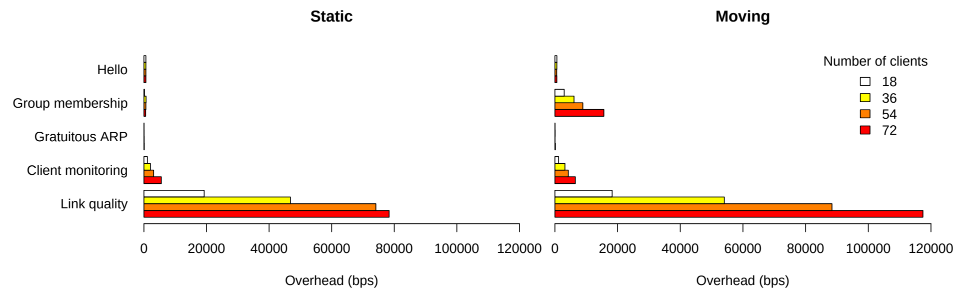


Figure 30: Overhead traffic.

We first measured the overhead traffic in tests with different numbers of stationary clients: 18, 36, 54 and 72. The clients were evenly spread in the mesh network, which corresponds to 1, 2, 3 and 4 clients, respectively, connected to each access point in our testbed. To support such experiments, which require a large number of simultaneous users, we implemented a client emulator that generates the appropriate control traffic associated with a regular client. From the 802.11 network and from the system's perspective, there was no difference between an emulated client and a real client in terms of control traffic. In the second test we evaluated the system while the clients were moving through the coverage area, each one randomly switching its access point about every minute.

Our measurement reflect the traffic seen by a single mesh node, node 11. For each type of traffic, we measured the overhead traffic considering the full size of the packets including the IP and UDP headers (Figure 30).

In the stationary tests, the highest bandwidth consumer was the link quality update traffic. The average throughput of Link Quality messages per second increased from 19,258 bps to 78,362 bps. The second worst consumer (although five times less) was the link monitoring traffic, which linearly increased from 1,106 bps to 5,564 bps. The rest of the traffic is low: as expected, the *hello* protocol has a constant overhead, which amounted to approximative 633 bps (1.16 average messages per second) while the traffic generated by joins/leaves stayed below 700 bps. The gratuitous ARP traffic was almost zero as the clients were stationary. Overall, the average overhead increased linearly with the number of clients, from 20.7 Kbps for 18 clients to 82.2 Kbps for 72 clients.

In the tests with moving clients, the highest bandwidth consumer continues to be link quality traffic − with a maximum of 117,436 bps for 72 clients. However the second is now the group membership traffic, which grows from 2,962 bps for 18 clients to 15,627 bps for 72 clients. This is because the movement of the clients resulted in increased activity on their *Client* and Data Groups. Since the clients moved randomly in the network, the density per node stays about the same throughout the experiments, therefore the link quality monitoring traffic is about the same as in the stationary test. The gratuitous ARP traffic is higher than before (each client experienced a handoff approximately every minute, which corresponds to more than one handoff per second in the entire network) but overall is extremely low. The network topology remained unchanged causing the same amount of hello traffic. Overall, the average overhead increased linearly with the number of clients, from 22.5 Kbps for 18 clients to 127.1 Kbps for 72 clients.

The aggregate management overhead increases linearly with the addition of clients, from 1.4 Kbps per client for stationary clients, to 1.9 Kbps per client for moving clients.

Finally, we demonstrate the operation of the system in a more diverse scenario, and show the overhead traffic sent and received by node 11 during the experiment (Figure 31). We started with no clients in the system (section A), and then gradually added 72 stationary clients evenly spread in the mesh network (section B). All the clients are stationary for some time (section C), after which half of them started to move (section D).

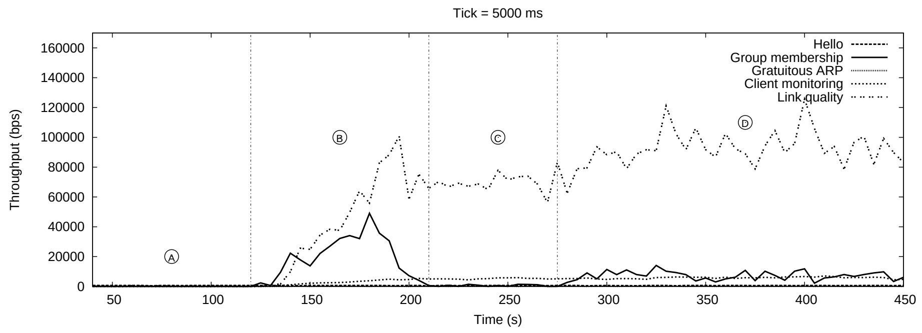


Figure 31: 18 nodes, 72 clients. Overhead Traffic. (A) no clients, (B) 72 clients connect, (C) all clients are stationary, (D) 36 of the clients start moving throughout the mesh.

Throughout the experiment, the hello and gratuitous ARP traffic stays very low, compared to other components. As clients join the network, we see a small increase in the client monitoring traffic, which remains stable after all the clients are connected. In contrast, as clients join the network, we see a significant increase in group state update traffic due to mesh nodes joining the *Control* and *Data* groups for the clients in their vicinity. Because the clients are stationary, this traffic goes back to zero after the updates are propagated in the network (section C). However, when some of the clients start to move (section D), the group state overhead traffic increases again as an effect of membership changes in the *Control* group (due to new clients coming within the vicinity of mesh node 11) and the *Data* group (due to handoffs). In the same way, link quality traffic increases while the clients join the network, but afterwards remains high since mesh nodes periodically share link quality information. We notice a small increase in this traffic when clients start to move (section D), mainly due to more clients coming within the vicinity of node 11.

7.3 Experiments summary

The experiments show that the SMesh protocols provide real-time handoff, with a low overhead caused by duplicates during periods of instability caused by handoffs. When sending and receiving both UDP and TCP traffic, the connections were not interrupted, and the loss when a mobile client roams was minimal.

As expected, a short interruption happens when the access point serving the client or its closest Internet gateway suddenly crashes. In such a cases, the system recovers, and within a few seconds is able to re-route packets through the network.

The management overhead of the mesh network grows linearly with the number of clients, in the worst case at a rate of about 2 Kbps per client. This overhead does not depend on the amount of data the mobile clients send or receive. As opposed to the wireless intra-domain overhead, which is only dependent on the number of clients, the inter-domain overhead is directly proportional to the number of connections each client has. Considering that the capacity of 802.11 wireless networks is in the order of tens of Mbps, we conclude that the management overhead of SMesh is reasonable.

8 Conclusion

This article presented the design, implementation and evaluation of SMesh, the first seamless 802.11 wireless mesh network with fast handoff that supports real-time applications, such as interactive Voice over IP, without any degradation in quality of service. The handoff is transparent for the mobile client, and the entire mesh network is seen by the client as a single, omnipresent access point.

Fast handoff is achieved by using multicast groups to coordinate decisions between access points and between Internet connected access points to seamlessly transfer connections as the mobile clients move throughout the mesh. We also optimized the use of the wireless medium by shortcutting wireless hops through wired connections.

We demonstrated the efficiency of our protocols through live experiments in a 18-node deployment. Our approach achieves very good results, allowing unmodified mobile clients to roam freely throughout the wireless coverage area of the mesh network without any interruption in service. We quantified the overhead and demonstrated that it is small compared to the data traffic.

The SMesh system is fully functional and available as open-source at www.smesh.org.

References

[1] Atul Adya, Paramvir Bahl, Jitendra Padhye, Alec Wolman, and Lidong Zhou. A multi-radio unification protocol for IEEE 802.11 wireless networks. In *BROADNETS '04: Proceedings of the First International Conference on Broadband Networks (BROADNETS'04)*, pages 344–354, Washington, DC, USA, 2004. IEEE Computer Society.

[2] S. Akyildiz, I.F.; Jiang Xie; Mohanty. A survey of mobility management in next-generation all-ip-based wireless systems. *Wireless Communications, IEEE*, 11(4pp):16–28, Aug 2004.

[3] Wang X. Akyildiz, I.F. and W. Wang. Wireless mesh networks: A survey. *Computer Networks Journal (Elsevier)*, Mar 2005.

[4] Yair Amir and Claudiu Danilov. Reliable communication in overlay networks. In *Proceedings of the IEEE DSN 2003*, pages 511–520, June 2003.

[5] Yair Amir, Claudiu Danilov, Stuart Goose, David Hedqvist, and Andreas Terzis. 1-800-OVERLAYS: Using overlay networks to improve VoIP quality. In *Proceedings of the ACM NOSSDAV 2005*, pages 51–56, June 2005.

[6] Yair Amir, Claudiu Danilov, Michael Hilsdale, Raluca Musaloiu-Elefteri, and Nilo Rivera. Fast handoff for seamless wireless mesh networks. In *MobiSys 2006: Proceedings of the 4th international conference on Mobile systems, applications and services*, pages 83–95, New York, NY, USA, 2006. ACM Press.

[7] Yair Amir, Claudiu Danilov, Raluca Musaloiu-Elefteri, and Nilo Rivera. An inter-domain routing protocol for multi-homed wireless mesh networks. *International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2007), Helsinki, Finland*, June 2007.

[8] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proc. of the 18th Symposium on Operating Systems Principles*, pages 131–145, October 2001.

[9] Yigal Bejerano, Israel Cidon, and Joseph (Seffi) Naor. Efficient handoff rerouting algorithms: a competitive on-line algorithmic approach. *IEEE/ACM Trans. Netw.*, 10(6):749–760, 2002.

[10] John C. Bicket, Daniel Aguayo, Sanjit Biswas, and Robert Morris. Architecture and evaluation of an unplanned 802.11b mesh network. In *MOBICOM*, pages 31–42, 2005.

[11] Milind M. Buddhikot, Adiseshu Hari, Kundan Singh, and Scott Miller. Mobilenat: A new technique for mobility across heterogeneous address spaces. *MONET*, 10(3):289–302, 2005.

[12] Ramon Caceres and Venkata N. Padmanabhan. Fast and Scalable Wireless Handoffs in Support of Mobile Internet Audio. *ACM Journal on Mobile Networks and Applications*, 3(4):351–363, 1998.

[13] J.D. Camp and E.W. Knightly. The ieee 802.11s extended service set mesh networking standard. *Communications Magazine, IEEE*, 46(8):120–126, August 2008.

[14] Joseph D. Camp, Edward W. Knightly, and William S. Reed. Developing and deploying multi-hop wireless networks for low-income communities. *Journal of Urban Technology*, 13(3):129–137, 2008.

[15] A.T. Campbell, J. Gomez, Sanghyo Kim, Chieh-Yih Wan, Z.R. Turanyi, and A.G. Valko. Comparison of ip micromobility protocols. *Wireless Communications, IEEE*, 9(1):72–82, Feb. 2002.

[16] Benjamin A. Chambers. The grid roofnet: a rooftop ad hoc wireless network. Master's thesis, Massachusetts Institue of Technology, May 2002., 2002.

[17] Carla-Fabiana Chiasserini. Handovers in Wireless ATM Networks: In-Band Signaling Protocols and Performance Analysis. *IEEE Transactions on Wireless Communications*, 1(1), Jan 2002.

[18] Yang-hua Chu, Sanjay G. Rao, and Hui Zhang. A Case For End System Multicast. In *Proceedings of ACM SIGMETRICS*, June 2000.

[19] D. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. *In Proceedings of MOBICOM 2003, San Diego*, 2003.

[20] CUWiN. The Champaign-Urbana community wireless network.

[21] S. Das, A. Mcauley, A. Dutta, A. Misra, K. Chakraborty, and S.K. Das. Idmp: an intradomain mobility management protocol for next-generation wireless networks. *Wireless Communications, IEEE*, 9(3):38–, June 2002.

[22] C. Demichelis and P. Chimento. IP Packet Delay Variation Metric for IP Performance Metrics (IPPM). *RFC 3393*, Nov 2002.

[23] Richard Draves, Jitendra Padhye, and Brian Zill. Comparison of routing metrics for static multi-hop wireless networks. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 133–144, New York, NY, USA, 2004. ACM.

[24] Richard Draves, Jitendra Padhye, and Brian Zill. Routing in multi-radio, multi-hop wireless mesh networks. In *MobiCom '04: Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 114–128, New York, NY, USA, 2004. ACM Press.

[25] R. Droms. Dynamic Host Configuration Protocol. *RFC2131*, Mar 1997.

[26] K. Egevang and P. Francis. The IP Network Address Translator (NAT). *RFC1631*, May 1994.

[27] S. Ganguly, V. Navda, K. Kim, A. Kashyap, D. Niculescu, R. Izmailov, S. Hong, and S.R. Das. Performance optimizations for deploying voip services in mesh networks. *Selected Areas in Communications, IEEE Journal on*, 24(11):2147–2158, Nov. 2006.

[28] A. Grilo, P. Estrela, and M. Nunes. Terminal independent mobility for ip (timip). *Communications Magazine, IEEE*, 39(12):34–41, Dec 2001.

[29] K. Malki H. Soliman, C. Castelluccia and L. Bellier. Hierarchical mobile ipv6 mobility management (hmipv6). June 2004.

[30] Ahmed A.-G. Helmy, Muhammad Jaseemuddin, and Ganesha Bhaskara. Multicast-based mobility: A novel architecture for efficient micromobility. *IEEE Journal on Selected Areas in Communications*, 2004.

[31] Robert Hsieh, Zhe Guang Zhou, and Aruna Seneviratne. S-MIP: A seamless handoff architecture for mobile IP. In *INFOCOM*, 2003.

[32] B. Liu, Z. Liu, and D. Towsley. On the capacity of hybrid wireless networks. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 2:1543–1552 vol.2, March-3 April 2003.

[33] Arunesh Mishra, Minho Shin, and William Arbaugh. An empirical analysis of the IEEE 802.11 MAC layer handoff process. *SIGCOMM Comput. Commun. Rev.*, 33(2):93–102, 2003.

[34] OpenWrt. OpenWrt. http://openwrt.org.

[35] C. Perkins. IP Mobility Support. *RFC2002*, Oct 1996.

[36] David C. Plummer. Ethernet Address Resolution Protocol: Or converting network protocol addresses to 48.bit Ethernet address for transmission on Ethernet hardware. *RFC826*, Nov 1982.

[37] Krishna N. Ramachandran, Milind M. Buddhikot, Girish Chandranmenon, Scott Miller, Elizabeth M. Belding-royer, and Kevin C. Almeroth. On the design and implementation of infrastructure mesh networks. In *in IEEE Workshop on Wireless Mesh Networks (WiMesh)*, 2005.

[38] Ishwar Ramani and Stefan Savage. Syncscan: Practical Fast Handoff for 802.11 Infrastructure Networks. In *Proc. of IEEE INFOCOM*, march 2005.

[39] R. Ramjee, T. La Porta, S. Thuel, K. Varadhan, and S.Y. Wang. Hawaii: a domain-based approach for supporting mobility in wide-area wireless networks. *Network Protocols, 1999. (ICNP '99) Proceedings. Seventh International Conference on*, pages 283–292, Oct.-3 Nov. 1999.

[40] S. Seshan, H. Balakrishnan, and R. Katz. Handoffs in Cellular Wireless Networks: The Daedalus Implementation and Experience. *Kluwer Journal on Wireless Personal Communications, 1996.*, 1996.

[41] S. Sharma, Ningning Zhu, and Tzi cker Chiueh. Low-latency mobile ip handoff for infrastructure-mode wireless lans. *Selected Areas in Communications, IEEE Journal on*, 22(4):643–652, May 2004.

[42] SMesh. The SMesh Wireless Mesh Network. http://www.smesh.org.

[43] Spines. The Spines Overlay Network. http://www.spines.org.

[44] Diane Tang and Mary Baker. Analysis of a Metropolitan-Area Wireless Network. *ACM/Kluwer Wireless Networks. Special issue: Selected Papers from Mobicom'99*, 8(2/3):107–120, 2002.

[45] V. Navda, A. Kashyap, S. Das. Design and evaluation of imesh: an infrastructure-mode wireless mesh network. In *6th IEEE WoWMoM Symposium,*, June 2005.

[46] András G. Valkó. Cellular ip: a new approach to internet host mobility. *SIGCOMM Comput. Commun. Rev.*, 29(1):50–65, 1999.

[47] Jon-Olov Vatn. An experimental study of IEEE 802.11b handover performance and its effect on voice traffic, 2003.

[48] H. Velayos and G. Karlsson. Techniques to reduce the ieee 802.11b handoff time. *Communications, 2004 IEEE International Conference on*, 7:3844–3848 Vol.7, June 2004.

[49] Hidetoshi Yokota, Akira Idoue, and Toru Hasegawa. Link layer assisted mobile ip fast handoff method over wireless lan networks. In *in Proceedings of ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, pages 131–139. ACM Press, 2002.