# Design and Development of a Management Solution for Wireless Mesh Networks based on IEEE 802.11s

Michael Rethfeldt, Peter Danielis, Guido Moritz, Björn Konieczek, Dirk Timmermann
University of Rostock
Institute of Applied Microelectronics and Computer Engineering
18051 Rostock, Germany, Tel.: +49 381 498-7269
Email: michael.rethfeldt@uni-rostock.de

***Abstract*—The broad availability of WLAN-capable off-the-shelf hardware lets WLAN mesh networks appear as promising technology for future distributed wireless applications. Featuring automatic device discovery, interconnection and routing, they provide a higher scalability, flexibility, and robustness compared to common centralized WLAN infrastructures. Besides these advantages, characteristics such as variable network topologies and link qualities imply new technical challenges for administration and real-world operation. Adopted in late 2011, IEEE 802.11s appears as new WLAN standard amendment, enabling vendor-independent mesh networks based on the widespread WLAN technology. However, network monitoring and management fall out of the standardization scope and are therefore not specified. In this paper, we present a novel 802.11s management solution based on the SNMP protocol. It covers dynamic mesh bootstrapping, error recovery, status monitoring and remote configuration. The presented solution was implemented and evaluated in a real-world testbed comprising more than 10 mesh nodes.**

## I. Introduction

Both the growing variety and affordability of mobile consumer devices serve as a catalyst for next-generation wireless services and applications. The vision of the "Internet of Things" (IoT) is characterized by the seamless integration of ambient embedded systems for user assistance [1]. Highly cooperative ensembles are built from a broad spectrum of devices such as sensors, actors, smart displays, and especially wearables like smart phones or tablets. This leads to complex networks that can provide distributed information services, e.g., in public places or smart offices. Further applications include home and building automation or the monitoring of industrial facilities. The widely-supported IEEE 802.11 WLAN (Wireless Local Area Network) standard family [2] is already omnipresent in today's home and office environments. Nevertheless, the currently prevalent centralized infrastructure mode is not well-suited for future applications in terms of scalability, flexibility, and robustness [3]. The network comprises a central access point (AP) and multiple stations (STAs) in direct radio range to the AP. The AP manages data forwarding between the associated STAs and therefore acts as single point of failure. Network extension is commonly achieved by a wired "Distribution System" (DS), e.g., via Ethernet. This leads to costly deployment if good coverage is needed.

In contrast, decentralized WLAN mesh networks are characterized by a flexible, scalable, and failsafe topology [3]. Distributed mesh routing is handled by all network nodes. Neighbors within radio range (peers) automatically interconnect and establish paths to selected targets. Thus, the network becomes more robust to changes in link availability and quality. An easy and economic network extension is simply possible by bringing in additional mesh nodes.

The higher dynamics and complexity of WLAN mesh networks put many challenges on monitoring and management, as no central AP is keeping track of the network status and associated devices. Moreover, all nodes must be initialized properly, operate on the same wireless channel, and share a common mesh configuration. In general, many manual configuration steps are required on each device. This is not suitable for a large number of nodes, and especially for complex mesh topologies, such as multi-radio / multi-channel setups [4]. Thus, mesh bootstrapping has to be performed in a distributed fashion where every node is equipped with self-configuration capabilities. Although a self-forming mesh network could already run unattended, the limited scope per node inherently limits optimization potentials. To regain a global network view, it is necessary to collect status data of all nodes [4]. Based on these information, it becomes possible to identify weak regions and misconfiguration, and to derive suitable optimization steps. Thereby, it is desirable to enrich a mesh node with monitoring and management capabilities. Furthermore, devices may not provide any user interface or may not be accessible when mounted on walls or roof tops. Therefore, it is necessary to provide remote control functionality to ensure the practical operation and maintainability of WLAN mesh networks. Consequently, we have developed a centralized management approach with a dedicated device role for remote monitoring and configuration. Dynamic network bootstrapping and error recovery are already performed in a decentralized manner by all mesh nodes.

The remainder of this paper is organized as follows: Section II outlines the basic principles of the IEEE 802.11s WLAN mesh standard, its Linux implementation *open80211s*, and the SNMP network management protocol. In Section III, we discuss related work in WLAN mesh management. Section IV describes the design of our mesh management framework. In Section V, we illustrate the results of our real-world testbed evaluation. Finally, we give a conclusion in Section VI and briefly state possible improvements and approaches for future research.

## II. Technological Basis

### A. IEEE 802.11s

In September 2011, IEEE 802.11s was ratified as the first industry WLAN mesh standard [2], [5]. As an amendment to the existing 802.11 specification, all mesh functionality is fully integrated into the existing MAC layer while the underlying physical layer (802.11 a/b/g/n/ac) remains untouched. Already existing MAC-layer frame types have been extended to enable mesh functionality such as automatic peering and routing (path selection). By reusing the existing MAC mechanisms, they are expected to perform with less overhead, compared to earlier, incompatible mesh routing protocols. The mesh topology becomes invisible to all higher layers, which avoids any modifications above the WLAN specification. Next to simple STAs with mesh capability, *Mesh Points* (MP), 802.11s also defines specialized nodes that act as gateways to external networks (*Mesh Portal* - MPP) or as AP for legacy devices (*Mesh Access Point* - MAP). Figure 1 shows an example 802.11s mesh topology.

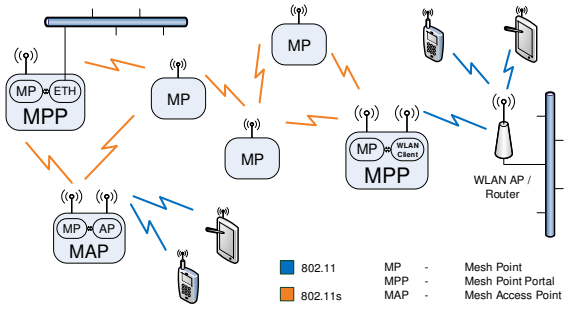


Fig. 1. Example 802.11s mesh topology

To ensure interoperability, the 802.11s standard specifies a mandatory basic mesh profile. It consists of the *Hybrid Wireless Mesh Protocol* (HWMP) for path selection and the radio-aware *Airtime Link Metric* (ALM). ALM represents costs for choosing a specific path in the mesh network by considering technology parameters of the physical layer and the wireless medium. Every mesh node is solely aware of its direct peers and nodes in multi-hop distance, to which communication has been explicitly initiated, e.g., on application level. A node's path table exclusively contains forwarding rules to target nodes over suitable next-hop peers, which are continuously updated. To every target node, only the best rule is kept, represented by the smallest ALM cost. Altogether, 802.11s dictates only a minimal mandatory mesh feature set. The use of HWMP inherits a limited network view on each node which renders network-wide management a difficult task. The mesh standard permits the use of vendor-specific path selection protocols that potentially entail a global network view per node. However, this approach would compromise overall interoperability. Thus, further management functions have to be implemented as separate solution.

### B. open80211s

The project *open80211s* [6] is a reference implementation of 802.11s for the Linux platform and used as basis for our solution. It is currently the most advanced 802.11s implementation and already satisfies all mandatory and various optional parts of the standard. The program code of *open80211s* is integral part of the WLAN stack inside the mainline Linux kernel.

### C. SNMP

The *Simple Network Management Protocol* (SNMP) has become a de facto standard for the management of IP-based LANs [7]. Its first version was specified in 1988 by the IETF. The latest iteration *SNMPv3*, released in 2002, was extended by complex security mechanisms. SNMP follows a client/server model consisting of manager (client) and agent (server). The SNMP agents, running on every network device, provide their local status information as *Managed Objects* (MOs). MOs are structured in a hierarchical *Management Information Base* (MIB). Every MO can be uniquely addressed by its *Object Identifier* (OID). Many standardized and enterprise MIBs are already available. With the reserved sub-tree "experimental", it is possible to define own MIB extensions, e.g., for research projects. Communication between client and agent is bound to UDP transport by default and follows a simple request-response principle. SNMP defines message types to request (GET), write (SET), and publish (TRAP) MO data, as well as numerous error codes to identify problems. MOs and data types are specified using ASN.1-based syntax rules. For transport, a binary encoding is used. Considering its wide acceptance, large set of features, and lightweight encoding, SNMP appears as a well-suited protocol also for the use in WLAN mesh networks.

## III. Related Work

Several management solutions have been proposed for IP-layer WLAN mesh protocols. Many approaches extended SNMP, supported it by proxy, or integrated it directly. As shown in earlier publications [8], [9], SNMP has been successfully applied in common WLAN infrastructures. Nevertheless, there is still a lack of solutions with particular focus on 802.11s and its characteristics. The UAVnet project [10] is the only 802.11s testbed we know of to explicitly include management functionality. The network is deployed by unmanned aerial vehicles (UAV), forming a wireless relay for disaster scenarios. Administration is based on the ADAM framework. It supports firmware deployment, logging per node, time synchronization, link loss recovery and provides a web interface to adjust device settings. Nevertheless, focus is not on mesh management and optimization but mainly on node initialization and flight control. ANMP [11], Guerrilla [12] and MannaNMP [13] present SNMP-based management of ad-hoc networks. Modified SNMP messages and self-defined MIB extensions are used for node monitoring. These works further describe concepts of hierarchical network clustering. Prabhu [14] describes a hierarchical concept for SNMP usage in ad-hoc networks proposing a division of specialized master- and sub-agents for different management tasks.

## IV. 802.11s Mesh Management Solution

We present a management solution specifically designed for 802.11s networks that addresses bootstrapping, monitoring,

and remote configuration. This includes MAC- and IP-layer auto-configuration to ensure node availability for further management tasks. We further provide autonomous error recovery to re-initialize a node after misconfiguration or failure. In contrast to link state routing protocols, HWMP induces a limited mesh topology knowledge per node. Thus, network monitoring is implemented above 802.11s to regain a global scope. Furthermore, standard-specific features (link blocking, path modification, etc.) are made remotely accessible. Our framework is realized as Java application on top of *open80211s* for Linux. Following a modular architecture, it comprises a larger platform-independent and smaller platform-dependent part. The latter serves as adapter for the underlying OS and can be easily replaced for future 802.11s implementations. Relying on SNMP, it provides the client-side *Mesh Manager* and server-side *Mesh Agent*. We use *SNMP4J* [15] to implement SNMP core functionality. Mesh status data and configuration interfaces are described as MIB extension. This way, interoperability with other SNMP-supporting management tools is ensured.

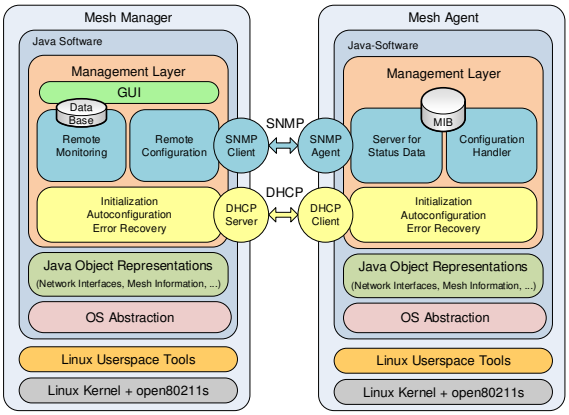


Fig. 2. Architecture of mesh management solution

As shown in Figure 2, the platform-specific part encapsulates user space tools for WLAN interface setup (*iw*), bridging (*brctl*), AP authentication (*hostapd*), IP and routing configuration (*ip*, *iptables*), DHCP functionality (*dnsmasq*, *dhclient*), and time synchronization (*ntpd*). Data extracted from the OS include general information like the list of current WLAN adapters, virtual interfaces, or IP routes as well as 802.11s-specific information. The "Peer List" and "Mesh Path List" hold status information for every link entry and path forwarding rule. While any link entry includes status, RSSI, sent and received bytes, or frame retransmission and failure count, any forwarding rule consists of next hop node, target node, and ALM metric [6]. Aggregation of the distributed link and path information allows it to derive a global network scope. Moreover, by considering information such as link bandwidth, RSSI, or ALM, it is possible to infer relative node distance and spatial mesh topology.

### A. Mesh Agent

The *Mesh Agent* core is built as state machine to handle dynamic initialization and self-configuration. This includes a detection of errors such as failing WLAN adapters and IP address loss. To be reachable via SNMP, at least one mesh interface must be connected at any time, both on 802.11s and IP level. Furthermore, a mesh node must not be isolated. Thus, its peer list is continuously checked for active links. Fall-back routines are triggered automatically on occurance of a *noNeighborTimeout* or *noIPAddrTimeout* to restart necessary initialization steps. A *Mesh Agent* traverses the states *INIT* (initializing mesh interface), *SCANNING* (scanning for surrounding mesh networks), *CONNECTED_L2* (connected on 802.11s MAC layer), and *CONNECTED_L3* (fully connected on IP level with SNMP agent running). Nodes automatically join the found mesh network with highest signal strength. If scan results are empty, a default profile is used. A DHCP client is started for IP address retrieval from a Manager, acting as DHCP server. When fully bootstrapped, an SNMP agent provides its status information (extracted from the OS) and remote configuration interfaces, specified as MIB extension. For Object Identifier (OID) assignment, we chose a free sub-tree in the *experimental* MIB branch (OID .1.3.6.1.3). We put "Status Data Objects" in **.experimental.1234.1* and "Configuration Objects" in **.experimental.1234.2*.

### B. Mesh Manager

The *Mesh Manager* role features the complementary SNMP client and DHCP server side to the Agent, as well as a Java GUI for status data visualization and remote configuration. Having joined a mesh network, a Manager checks if it is already managed, indicated by a successful DHCP request. Since multiple Managers are not supported yet, this is repeated until an unmanaged network is found. On success, the Manager periodically sends SNMP GET requests for all "Status Data Objects" to all available Agents. Long-term database storage enables the calculation of statistics and derivation of optimization steps out of the recorded network history. Remote configuration is performed by modifying a Mesh Agent's "Configuration Objects" through SNMP SET requests. Possible commands include wireless channel selection, node type changes (MP, MPP, MAP), path modifications, peer blocking, and the tuning of further 802.11s-specific parameters related to path selection or link establishment.

## V. Real-World Testbed Evaluation

We established a single-channel real-world 802.11s mesh testbed to show the practical suitability of our management solution. It comprises 10 ARM-based devices (1 x FOX Board G20, 7 x Raspberry Pi, 2 x Pandaboard [16]–[18]) as Agents and 1 notebook as Manager ((2 x 1,3 GHz Pentium SU4100, 4 GB RAM). All devices run a Linux OS (Debian 7, Ubuntu 12.04) with kernel v3.16-1 and use an 802.11g WLAN USB stick depending on the *rt2800usb* driver [19], [20].

First, we determined the maximum achievable UDP data rate for different hop counts to classify monitoring overhead. Only the notebook and the Raspberry Pis were used, all operating on channel 1 using default mesh parameters [6]. We used the "Peer Link Block" option to blacklist certain peers of a node and create a multi-hop path. For every path length,

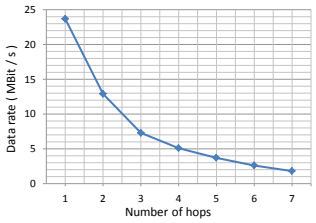


Fig. 3. UDP throughput

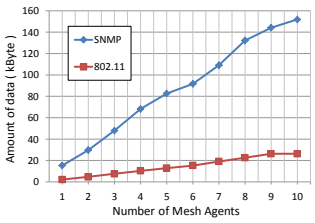


Fig. 4. SNMP overhead

10 UDP throughput measurements between Manager and last Agent were performed using *iperf* v2.0.5 [21]. Measurements were run for 10 seconds with a target rate set to 100 MBit/s. For each path length, results were averaged, as shown in Figure 3. For 1 hop, the UDP net data rate was 23.7 MBit/s. After 2 hops, it dropped to 12.9 MBit/s. For 7 hops, an average of only 1.7 MBit/s was measured. This trend could be expected, as every hop requires channel access and transmission time.

As a second step, we evaluated the cyclic SNMP monitoring overhead for 1 to 10 active Agents in 1-hop distance to the Manager. SNMP traffic was captured using Wireshark v1.10 (UDP datagrams without header) [22]. Separately, 802.11 control frames were captured for direct comparison. The query interval was set to 15 seconds. In each cycle, 10 MO tables were collected from each node. 40 cycles were averaged for every node count and then normalized to 1 cycle. Figure 4 shows the results. As expected, monitoring overhead scales linearly with node count and query interval. For 10 active Agents, only an average of 150 kB data is generated per query cycle. For an interval of 15 seconds and 10 Agents, the average rate is 80 kBit/s. Compared to the achievable single-hop UDP data rate (23.7 MBit/s) in the unmonitored network, this is an overhead of only 0.3 %.

In contrast to the single-hop setup, in a multi-hop scenario SNMP traffic will be forwarded by intermediary nodes and therefore induce a higher overhead. Nevertheless, the maximum considered size of a single-channel 802.11s network has been denoted as 50 nodes [23]. Thus, a realistic upper limit for path lengths can be expected. In future research, we will investigate multi-hop setups of different size.

In general, orthogonal WLAN channels between node pairs on a path can be used to increase throughput by reducing channel accesses and interference. Grouping of nodes into monitoring clusters can be a measure to keep path lengths down and prevent forwarding overhead. Additionally, SNMP supports TRAP messages for notifications that can be used to trigger a query cycle on events.

## VI. Conclusion

In this paper we present a Java-based management framework, tailor-made for 802.11s WLAN mesh networks. Our solution provides self-configuration and error recovery capabilities on top of 802.11s and further implements SNMP-based monitoring and remote configuration of the distributed nodes. Thereby, it regains a global network view that is otherwise lost due to a limited network scope, induced by HWMP path selection. Our 802.11s-specific MIB extension already covers the essential status data and configuration functions, needed for practical mesh operation and optimization. For our framework to become as failsafe as the underlying 802.11s network, in a next step the currently centralized Manager role will become decentralized. In future research we will use our framework to explore cross-layer optimization strategies within the boundaries of the 802.11s standard.

## Acknowledgment

The authors would like to thank the German Research Foundation (DFG), Research Training Group 1424 (MuSAMA) for their financial support.

## References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.
[2] "IEEE Standard for Information technology - Telecommunications and information exchange between systems Local and metropolitan area networks - Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, 2012.
[3] M. Portmann and A. A. Pirzada, "Wireless mesh networks for public safety and crisis management applications," *Internet Computing, IEEE*, vol. 12, no. 1, pp. 18–25, 2008.
[4] J. L. Duarte, D. G. Passos, R. L. Valle, E. Oliveira, D. C. Muchaluat-Saade, and C. V. N. de Albuquerque, "Management issues on wireless mesh networks," in *LANOMS*. IEEE, 2007, pp. 8–19.
[5] G. R. Hiertz, D. Denteneer, S. Max, R. Taori, J. Cardona, L. Berlemann, and B. Walke, "IEEE 802.11s: The WLAN Mesh Standard," *Wireless Commun.*, vol. 17, no. 1, pp. 104–111, Feb. 2010.
[6] "open80211s." [Online]. Available: http://open80211s.org/open80211s/
[7] R. Presuhn, "Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)," RFC 3416 (INTERNET STANDARD), Internet Engineering Task Force, Dec. 2002. [Online]. Available: http://www.ietf.org/rfc/rfc3416.txt
[8] R. Johnson, *Evaluating the use of SNMP as a wireless network monitoring tool for IEEE 802.11 wireless networks.* ProQuest/UMI, 2011.
[9] C. M. Vancea and V. Dobrota, "Snmp agent for wlan networks," *8th RoEduNet International Conference 'Networking in Education and Research', Galati, Romania*, 2009.
[10] S. Morgenthaler, T. Braun, Z. Zhao, T. Staub, and M. Anwander, "Uavnet: A mobile wireless mesh network using unmanned aerial vehicles," in *Globecom Workshops (GC Wkshps), 2012 IEEE*. IEEE, 2012, pp. 1603–1608.
[11] W. Chen, N. Jain, and S. Singh, "Anmp: Ad hoc network management protocol," *Selected Areas in Communications, IEEE Journal on*, vol. 17, no. 8, pp. 1506–1531, 1999.
[12] C.-C. Shen, C. Jaikaeo, C. Srisathapornphat, and Z. Huang, "The guerrilla management architecture for ad hoc networks," in *MILCOM 2002. Proceedings*, vol. 1. IEEE, 2002, pp. 467–472.
[13] F. A. Silva, L. B. Ruiz, T. R. M. Braga, J. M. S. Nogueira, and A. A. F. Loureiro, "Defining a wireless sensor network management protocol," in *LANOMS*, 2005, pp. 39–50.
[14] V. Prabhu H, "Master Subagent Based Architecture to Monitor and Manage Nodes in Mobile Ad-Hoc Networks," *International Journal of Engineering Research and Applications (IJERA)*, vol. 2, no. 3, pp. 1461–1465, May 2012.
[15] "Snmp4j." [Online]. Available: http://www.snmp4j.org/
[16] "ACME Systems." [Online]. Available: http://www.acmesystems.it/
[17] "Raspberry Pi." [Online]. Available: http://www.raspberrypi.org/
[18] "Pandaboard." [Online]. Available: http://pandaboard.org/
[19] "Buffalo AirStation N150 WLI-UC-GNM." [Online]. Available: http://www.buffalotech.com/products/wireless/usb-adapters/airstation-n150-wireless-usb-adapter
[20] "rt2x00 Project." [Online]. Available: http://rt2x00.serialmonkey.com
[21] "iperf." [Online]. Available: http://sourceforge.net/projects/iperf/
[22] "Wireshark." [Online]. Available: http://www.wireshark.org/
[23] J. Hauser, D. Baker, and W. S. Conner, "Draft PAR for IEEE 802.11 ESS Mesh," IEEE P802.11 Wireless LANs, Document 11-04/0054r2, Jan. 2004.