5G PPP
PUBLIC-PRIVATE PARTNERSHIP

European Commission | Horizon 2020 European Union funding for Research & Innovation

Dynamically Reconfigurable Optical-Wireless Backhaul/Fronthaul with Cognitive Control Plane for Small Cells and Cloud-RANs

D4.11 Wireless backhauling using Sub-6 GHz systems

This project has received funding from the European Union's Framework Programme Horizon 2020 for research, technological development and demonstration

Advanced 5G Network Infrastructure for the Future Internet

Project Start Date: July 1st, 2015 Duration: 36 months

H2020-ICT-2014-2 671551 December 31st, 2016 – Version 1.0

Project co-funded by the European Commission
Under the H2020 programme

Dissemination Level: Public

Grant Agreement Number:	671551
Project Name:	Dynamically Reconfigurable Optical-Wireless Backhaul/Fronthaul with Cognitive Control Plane for Small Cells and Cloud-RANs
Project Acronym:	5G-XHaul
Document Number:	D4.11
Document Title:	Wireless backhauling using Sub-6 GHz systems
Version:	2.0
Delivery Date:	December 31st, 2016
Responsible:	**I2CAT**
Editor(s):	Eduard Garcia-Villegas (**I2CAT-UPC**)
Authors:	Eduard Garcia-Villegas (**I2CAT-UPC**), Ilker Demirkol (**I2CAT-UPC**), Daniel Camps (**I2CAT**), August Betzler (**I2CAT**), Matteo Grandi (**I2CAT**), Josep Paradells (**I2CAT-UPC**).
Keywords:	IEEE 802.11, Wi-Fi, Sub-6 GHz, Software Defined Networking (SDN)
Status:	Final
Dissemination Level	PU
Project URL:	http://www.5g-xhaul-project.eu/

Table of Contents

List of Figures

List of Tables

Executive Summary

The next generation of mobile wireless networks, widely known as 5G, represents ambitious challenges for researchers of many fields. The 5G-XHaul project is focused on developing a new architecture capable of providing backhaul and fronthaul services to future 5G networks. 5G-XHaul not only has to cope with increased traffic demands of new access network technologies, but it also has to provide flexibility to accommodate different (and varying) services to one or multiple operators. 5G-XHaul's data plane achieves this flexibility by means of a converged optical and wireless architecture while, in the control plane, it follows the SDN paradigm.

The wireless part of the architecture encompasses both mmWave and Sub-6 GHz radio technologies to build multi-hop wireless mesh islands connecting the access network to the core. These technologies have the potential to offer the needed flexibility to, for example, allow an intelligent control plane to reconfigure the topology and data paths according to diverse criteria (e.g. capacity-oriented, maximize energy savings, react to broken links, etc.), accommodate flows with different QoS requirements (and honour different SLAs), allow multi-tenancy, etc. However, transport nodes participating in the wireless backhaul need to be adapted to the desired architecture in order to unleash said potential.

Wireless transport nodes must therefore expose different programmable features to the SDN-based control plane operating in 5G-XHaul. In this document, we define how IEEE 802.11-based Sub-6 GHz transport nodes are integrated within the 5G-XHaul framework, including the design of the node architecture and the programmable capabilities it makes available to the SDN control plane. The document also provides an evaluation of the system-wide benefits obtained when an intelligent control plane is able to orchestrate the defined capabilities. Finally, we propose a definite hardware platform whereupon we implement the 5G-XHaul Sub-6 GHz node and provide initial performance measurements.

1 Introduction

The present document is framed under the 5G-XHaul project. The major goal of the 5G-XHaul is to define a sound solution for the provision of wide band connectivity to any potential 5G radio access technology (RAT). 5G-XHaul is therefore a backhaul (BH) and fronthaul (FH) infrastructure that meets the requirements of future 5G networks. This can be achieved by means of a converged wireless and optical network, as the one shown in Figure 1-1. The converged network provides the flexible infrastructure capable of supporting a diverse and time-varying set of services [1].

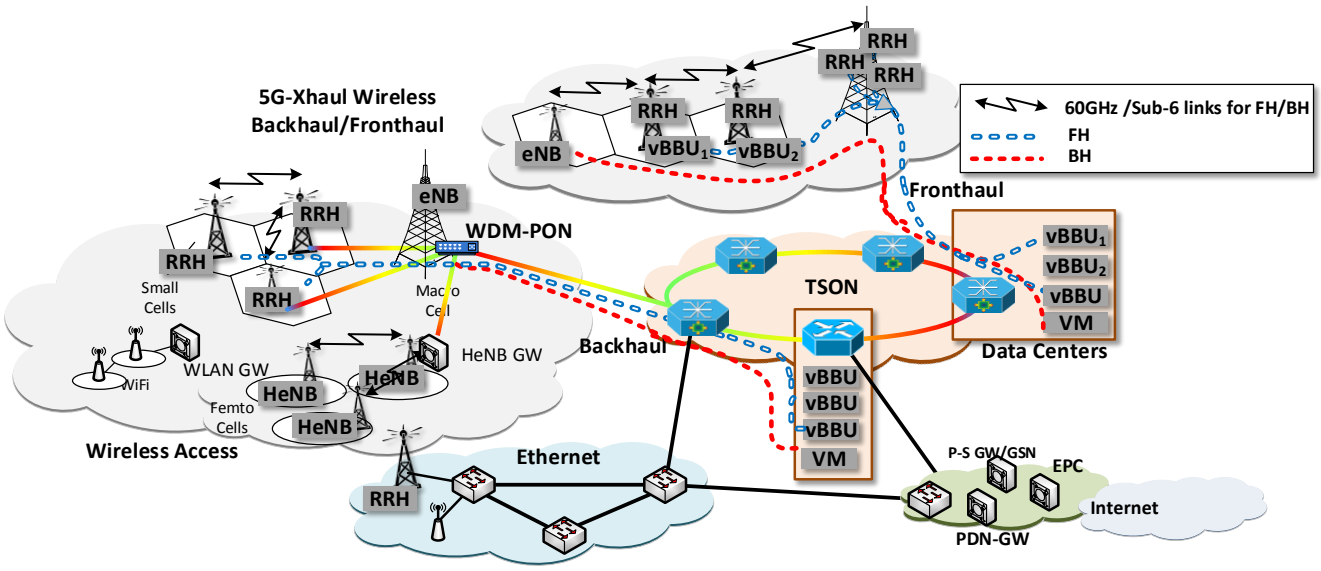


Figure 1-1: 5G-XHaul Architecture.

The wireless elements of the proposed infrastructure include both millimetre-Wave (mmWave) and Sub-6GHz technologies, which, combined, offer a good balance between bandwidth, range and flexibility. That flexibility enables a complete set of transport network configurations, summarized in Figure 1-2, and including i) mmWave FH with point-to-point (P2P) links; ii) Sub-6 GHz and mmWave multihop BH mesh, and iii) joint access/BH mesh for mmWave and Sub-6 GHz.

To support the wide variety of configurations defined in [1], the control plane must have access to a comprehensive set of parameters to rule the behaviour of the wireless BH/FH - or even Radio Access Network (RAN) nodes. This document describes the internal architecture of Sub-6 GHz nodes that enables such control. It also studies the integration of those nodes within the unified control plane based on Software Defined Networking (SDN), as envisioned by the 5G-XHaul and, through different proofs of concept, it also shows system-wide benefits obtained when an intelligent control plane is able to orchestrate the defined capabilities. Finally, a Linux-compatible hardware platform, equipped with multiple IEEE 802.11-compliant interfaces, is proposed as the basis upon which we can implement the described node architecture.

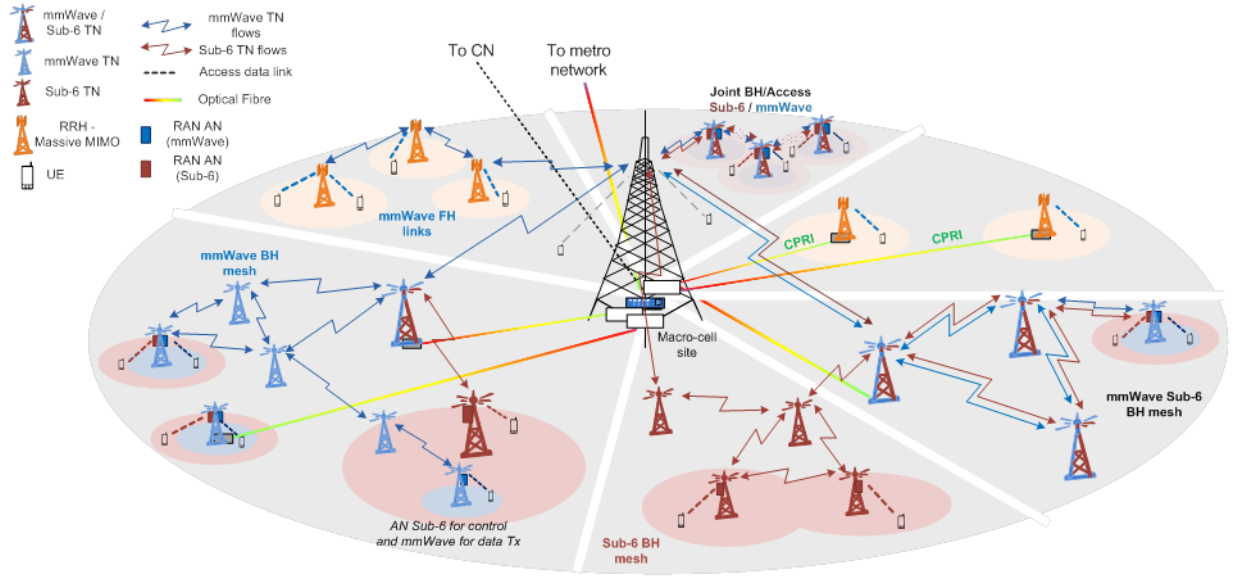


Figure 1-2: 5G-XHaul Wireless Connectivity Scenarios.

The document is structured as follows; after this introduction, section 2 provides an overview of the internal architecture proposed for the 5G-XHaul's Sub-6 GHz nodes and what interfaces they expose to the control plane. Section 3 details the programmable capabilities made available in those Sub-6 GHz backhaul nodes: remote monitoring and time division multiple access (TDMA) slot configuration. In section 4, we provide further details of the implementation of the features defined in sections 2 and 3, and evaluate the feasibility of different applications of such features as a proof of concept. After defining and evaluating the potential of the node architecture, in section 5 we finally propose and evaluate a definite hardware platform whereupon the 5G-XHaul Sub-6 GHz node is built. Concluding remarks are then discussed in section 6.

2 Design of Sub-6 GHz nodes

As already stated in [1], within the 5G-XHaul project, we focus our study of the Sub-6 GHz wireless BH on IEEE 802.11-based technologies (i.e. Wi-Fi). Its prominent position as non-line-of-sight (NLoS) RAT for data communications in small cells, along with its continuous evolution, make the IEEE 802.11 family of standards a good candidate to play a key role in future 5G deployments. Considering the latest advances in the physical layer (i.e. aggressive modulations up to 1024-QAM, MIMO, etc.), Wi-Fi is ready to offer throughput in the Gbps scale, which will surely grant it a place among the most widespread 5G RATs. However, its presence in the BH requires non-trivial adaptations, as discussed in this document. Wi-Fi networks use to operate in a distributed fashion, thus hampering its integration within a centrally controlled infrastructure. In this section, we deal with centralized management of Wi-Fi-based Sub-6 GHz transport nodes (TNs) by defining a novel architecture that enables the application of SDN techniques to control their operation. To that aim, IEEE 802.11 MAC and PHY are extended to enable remote programmability so that the 5G-XHaul control plane will be able to measure the state of the Sub-6 BH, while controlling forwarding rules and MAC transmission parameters. We would like to highlight at this point that the proposed extensions are based on generic Linux stack modules, avoiding specific driver-dependent modifications whenever possible. In this way, these 5G-XHaul Sub-6 GHz TNs can be deployed using a wide variety of (Linux-compatible) hardware platforms, such as the one presented in section 5.

2.1 Overall architecture

5G-XHaul's Sub-6 GHz BH nodes are equipped with one or more wireless interfaces (i.e. IEEE 802.11-based) to be managed by an SDN controller. In the proposed architecture, the neighbouring BH nodes that are reachable from a single wireless interface are abstracted inside the transport node as virtual Ethernet interfaces, which appear to the SDN controller as Ethernet ports connecting those wireless switches. Figure 2-1 exemplifies such abstraction, where four Sub-6 GHz transport nodes are connected using three non-interfering radio channels (red, blue and green): left part of the figure represents physical connections and the right part represents the abstracted topology as it would be read from the controller, where the three broadcast channels are interpreted as five different links (*L1* to *L5*).

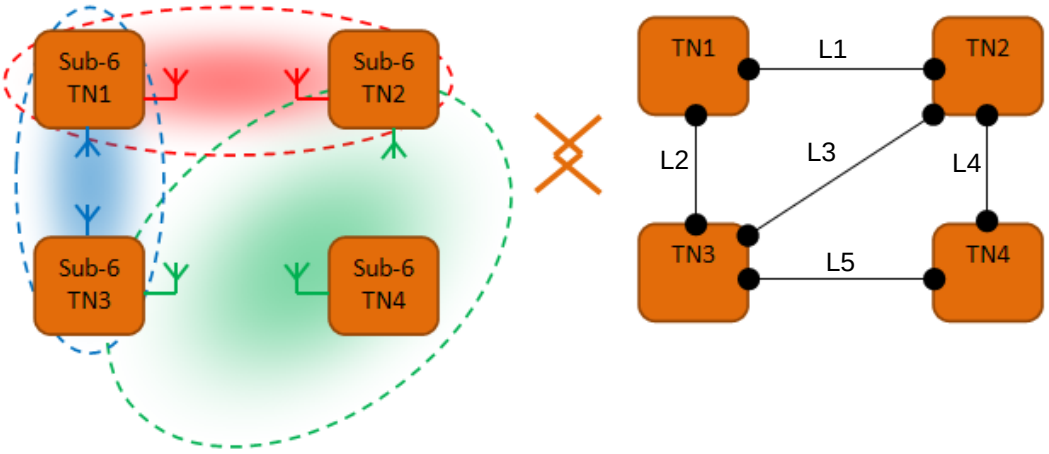


Figure 2-1: Abstraction of wireless transport nodes as SDN-enabled switches.

Thus, the SDN controller can control forwarding across the wireless BH in the same way as in a traditional wired network. In addition, this technology allows the transport switch to obtain, for each wireless port, real time measurements about the relevant wireless metrics, which are then reported to the SDN controller (cf. Section 3). Hence, the SDN controller is able to proactively optimize network resources while taking into consideration the conditions of the wireless channel. Figure 2-2 illustrates the architecture of a wireless BH node where we can observe the following major components:

- One or more wireless devices (***D***) inside a wireless transport node controlling access to the wireless medium in each wireless switch, where each wireless device ***D*** can be configured to operate independently.

- One SDN agent (***A***) controlling the forwarding plane in the wireless switch and communicating with the remote SDN controller (***C***), through the OpenFlow-based [2] Ext_{SB} interface.
- A functional entity (***MUX***), which multiplexes multiple wireless links over a single wireless device ***D***. For example, in Figure 2-2, the wireless device operating in channel *X* is in communication range of two other devices, hence the MUX de/multiplexes two virtual interfaces from/into one physical radio interface.
- A pair of interfaces Int_{DPi} and Int_{CP} running between the SDN agent ***A*** and wireless device ***D***, where: i) Int_{DPi} transports the data plane packets to be transmitted over the wireless network, and ii) Int_{CP} is a control plane interface that allows the SDN agent to poll the wireless device ***D*** about the instantaneous radio conditions in the network.

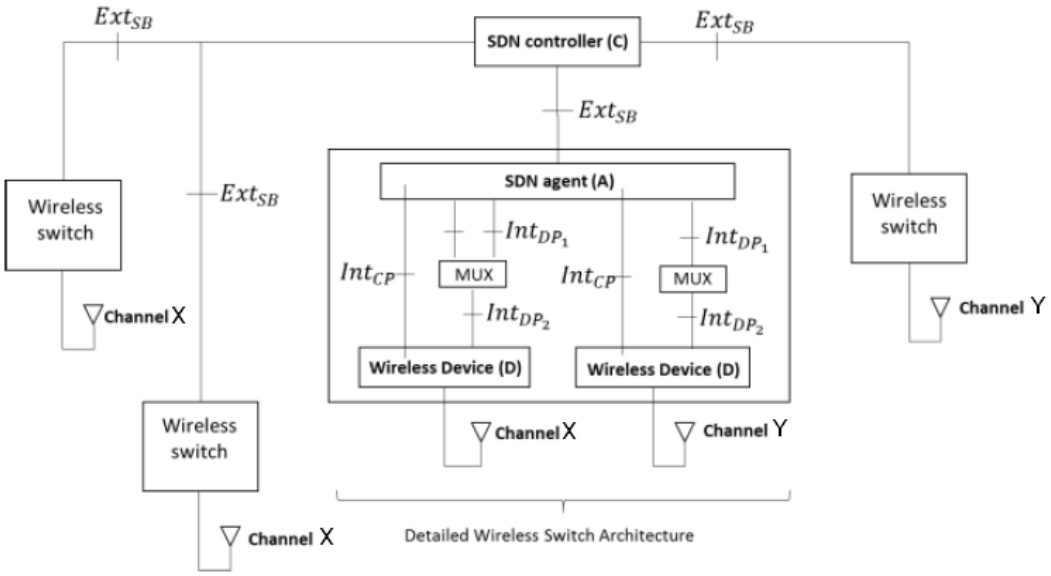


Figure 2-2: Internal logical architecture of Sub-6 GHz wireless transport nodes.

2.2 Functions and interfaces

Given that Ext_{SB} is implemented using OpenFlow, the SDN agent ***A*** is essentially an OpenFlow switch. OpenFlow switches control forwarding through a set of Ethernet ports that compose the switch datapath, where each port provides a physical point-to-point (P2P) connection to another switch in the network. Note, however, that our Sub-6 GHz TNs use Wi-Fi instead of Ethernet interfaces. Being the Wi-Fi channel a broadcast medium, each port could be used to connect to several neighbouring switches (e.g. green interfaces in Figure 2-1). However, considering that physical point-to-multipoint (P2MP) connectivity in an OpenFlow network is potentially disturbing to existing SDN software components[1], we hide in the proposed architecture the P2MP capabilities of wireless devices from the SDN agent through the introduction of the ***MUX***.

The ***MUX*** is a functional entity in charge of multiplexing data packets coming from the various Int_{DP1} interfaces over a single Int_{DP2} interface, and de-multiplexing them in the reverse direction. Due to its widespread availability, we use 802.1Q VLANs as a multiplexing mechanism. Notice that embedding a 4Byte VLAN tag in the data packets transmitted over the air incurs an additional overhead that is considered acceptable given the provided functionality. In addition, a VLAN tag allows multiplexing up to 4096 different Int_{DP1} interfaces, which is more than enough in our scenarios of interest (i.e. less than four peers per BH node). Thus, upon receiving a packet from an Int_{DP1i} interface, the ***MUX*** pushes the $VLAN_i$ tag to the packet and forwards the packet through Int_{DP2}. On the other hand, upon receiving a packet from the Int_{DP2} interface, the ***MUX*** pops the packet's $VLAN_i$ tag and forwards it to the appropriate Int_{DP1i} interface. Hence, VLANs encode the forwarding information. Next section 2.2.1 provides a detailed explanation about the way VLAN tags are agreed upon among wireless nodes. Finally, the interface Int_{DP2} carries VLAN tagged packets to the wireless device ***D***, which has been modified to

[1] many applications (e.g. LLDP discovery) heavily rely on the fact that physical links provide point to point connections.

interpret the VLAN tag and accordingly identify the neighbouring switch to which the packet must be transmitted.

2.2.1 VLAN management

Given that VLAN tags are used in the MUX to multiplex data packets coming from the SDN agent, and de-multiplex packets from the wireless device, a mechanism is required to derive the appropriate VLAN tags to be used to represent each Int_{DP1} interface. In particular the designed mechanism should fulfil the following requirements:

- Upon receiving a packet from the *MUX* with $VLAN_i$, the wireless device *D* should be able to decode the destination neighbouring switch for the packet, i.e. forwarding information is encoded in the VLAN identifier.
- Upon receiving a packet from the wireless device *D* with $VLAN_i$, the *MUX* should be able to decode the Int_{DP1} interface the packet should be forwarded to. Notice that the *MUX* receives an Ethernet frame, i.e. the wireless header is strip by the wireless device *D* and, thus, the *MUX* is unaware of the address of the wireless switch that transmitted this packet.

Similar problems exist, for instance, in Multiprotocol Label Switching (MPLS), where a Label Distribution Protocol (LDP) is used to distribute labels among nodes. Thus, a similar concept is used in our architecture, but instead of defining a new label distribution protocol we build upon the Peer Link Management (PLM) protocol available in any wireless device supported by the Linux kernel. PLM is a protocol defined as part of the 802.11s standard [4] (mesh extensions for 802.11), which allows wireless devices to establish a logical link with other neighbouring devices configured to operate in the same wireless network. PLM works by monitoring the Beacon frame transmissions from neighbouring devices, and implementing a two-way handshake where devices exchange wireless capabilities and a locally unique identifier for the peer link, known as Local Link IDentifier (LLID). Thus, after the PLM handshake, a wireless device obtains a unique local identifier for the peer link (LLID), and the LLID value used by the neighbouring device to refer to the same link, which is locally stored as the Peer Link ID (PLID). A simplified[2] example of the operation of the PLM protocol is depicted in the left part of Figure 2-3, where we can see how each wireless device *D* identifies potential neighbouring links with a pair of LLID/PLID values.

Our system builds on the exchanged LLID/PLID values to derive the VLAN tags used to multiplex several Int_{DP1} interfaces over a single wireless device. For this purpose, the MUX needs to be aware of the LLID/PLID values collected by the wireless device, which is possible using management tools for wireless interfaces available in the Linux kernel [8].

The operation of the *MUX* setup is illustrated through an example in the right part of Figure 2-3. As we can see in the figure, the *MUX* uses the LLID[3] value assigned by the wireless device to multiplex and de-multiplex the transmitted and received packets. In addition, upon having a new packet to transmit, the wireless device *D* inspects the packet's VLAN tag, uses the contained LLID value to discover the neighbouring device the packet needs to be transmitted to, and finally overwrites the carried VLAN value to the corresponding PLID for that link. Notice that the wireless device *D* needs to swap the VLAN value to PLID because PLID is the identifier for that link assigned by the device we are transmitting the packet to and, therefore, it is ensured to be unique in the receiving device. Finally, notice that every packet received by the wireless interface is forwarded up to the *MUX* and the SDN agent where the forwarding intelligence is located.

[2] Actually, for each link each device must initiate a two-way handshake but, for simplicity, only one is shown in Figure 2-3.
[3] The wireless device *D* is modified to only assign LLIDs between 0 and 4096, which is the maximum VLAN ID value.

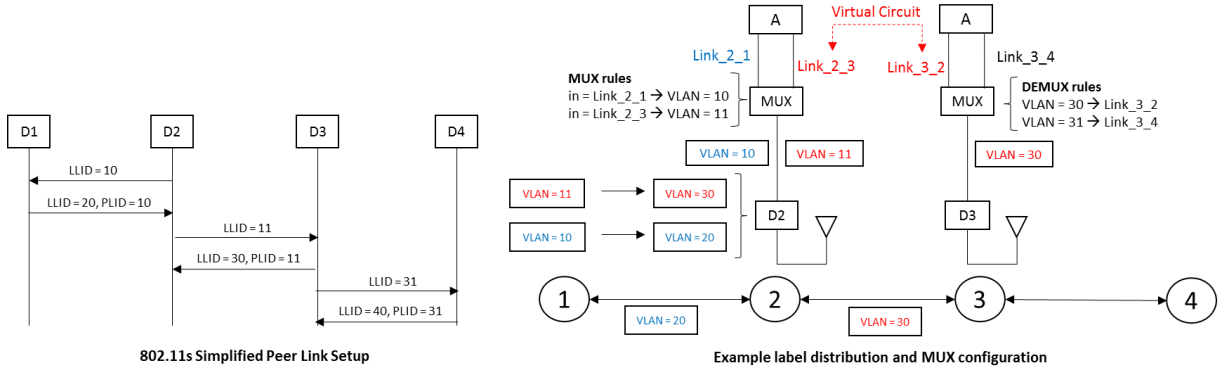


Figure 2-3: Example VLAN distribution mechanism.

2.2.2 In-band signalling

In OpenFlow-enabled networks, it is a common practice to use a separate infrastructure to carry the control connections (i.e. signalling between OpenFlow controller and managed switches) and user data. Notice though, that this is not possible in our scenarios of interest, where only a few TNs will have an additional wired connection. Therefore, in general, the signalling connection needs to be transported through the same wireless interfaces that form the datapath. Such configuration is referred to as an *in-band* signalling. In-band connectivity, though, poses a bootstrapping problem; that is, an initial configuration is required to find a path from the managed node to the controller since, under normal operation, the switch tables are initially empty and can only be populated after establishing the OpenFlow channel with the controller.

The bootstrapping problem can be solved either by implementing an automatic topology discovery mechanism or by providing an initial, pre-computed configuration (as detailed in section 4.1).

3 Programmable capabilities of Sub-6 GHz TNs

This section details the programmable capabilities made available in our Sub-6 GHz BH nodes: remote monitoring and TDMA slot configuration. The former includes periodic statistics reporting by the Sub-6 GHz TNs as well as event-triggered notifications upon link failure detection. The latter allows the centralized scheduling of TDMA-like access on IEEE 802.11-based TNs, which is useful to provide a fair share of resources between the access and the backhaul networks and also to manage QoS with greater precision than legacy IEEE 802.11's CSMA.

3.1 Network state monitoring

The SDN controller ***C*** collects statistic reports from the agent nodes, which include information about their wireless ports, internal state (e.g. CPU load), and information about the data flows traversing them. This information is used to get a holistic view of the system, and to trigger corrective actions when needed, that is, when the controller detects a precarious network state (e.g. link congestion). Those adjustments may include any of the following actions:

- Transmitted power assignments: increase power to enable faster modulation and coding schemes (MCS), or reduce it to minimize interference over neighbouring links.
- Frequency channel management: to reduce interference with neighbouring links and minimize external interference (in case of license-free bands).
- Carrier sense threshold adjustments: adjust CSMA parameters in dense environments.
- Resource reallocation: modify bandwidth, MCS or time slot assignments so that link capacity is adjusted to carried traffic.
- Flow reallocation: reallocate data flows through different paths to avoid congested (or lossy) links or to recover data paths after a link/node failure.

In order to effectively apply the aforementioned corrective actions, the SDN controller ***C*** in Figure 2-2 needs not only the typical per-port and per-flow statistics (e.g. number of packets, Bytes, errors, etc.) supported by most of current SDN schemes, but also ***C*** requires information about the radio conditions. In the proposed architecture, two interfaces are used to bring radio information to the controller. Firstly, the interface Int_{CP} allows the SDN agent ***A*** in the wireless switch to retrieve measured radio conditions from the wireless device ***D***. Secondly, the interface Ext_{SB} is used to transport the collected radio parameters from the SDN agent to ***C***. Thus, the SDN agent may use at any time interface Int_{CP} to poll the wireless device for the latest radio conditions. The implementation of this interface is described in section 4.2. Finally, the Ext_{SB} interface is implemented using OpenFlow [2], which has been extended to transport radio parameters.

3.1.1 Statistics collected by each Sub-6 GHz TN

Each Sub-6 GHz TN maintains information about its internal state, its wireless network interfaces and ports, and the observed network environment. The set of statistics gathered by each agent *A* is divided in four categories: interface statistics, port statistics, device statistics and flow statistics.

Table 3-1 provides a summary of some of the most relevant parameters, including those radio statistics added as an extension to OpenFlow (the complete list of statistics reported by a standard OpenFlow switch can be found in [2]). Those statistics are then reported periodically to the controller ***C*** upon request.

Table 3-1: Summary of relevant statistics reported by Sub-6 GHz TNs.

Statis-tics	Parameter	Definition
Interface Statis-tics	Channel usage	Portion of time ($0 < u < 1$) that the channel is reported to be busy.
	Channel number	Operational channel of a wireless interface.
Port Statis-tics	Signal strength	RSSI measured on the last packet received from the peer TN in this port.
	Avg. signal strength	Average RSSI of packets received from the peer TN in this port.
	TX/RX bitrate	PHY bitrate of the last transmitted/received packet in this port.
	TX/RX packets	Number of packets sent/received through this port (measured between layer 2 and layer 3).
	TX/RX bytes	Number of bytes sent/received through this port (measured at layer 2).
	TX retries	Cumulative retry counts.
	TX failed	Number of frames dropped due to excessive retries on the wireless channel.
	Queue size	Level of occupancy of the different transmission queues.
	TX/RX errors	Number of TX/RX errors detailed by error source (frame alignment, CRC, collision, etc.).
Device Statis-tics	CPU load	Average CPU load in % measured in this device.
Flow Statis-tics	Duration	Time since the flow was installed in the switch.
	Priority	Priority of that entry
	Timeout	Duration in seconds before expiration of that entry.
	Packet count	Number of packets in a given the flow.
	Byte count	Number of bytes in a given the flow.

3.1.3 Notification of broken link

When communication between two neighbouring Sub-6 GHz nodes over a given link is no longer possible (e.g. due to unbearable interference, hardware failure, mobility of the nodes, etc.), the network should be capable of self-healing so that the affected flows are promptly reallocated through new paths, when possible. Therefore, detection and notification of broken links is another feature that should be present in the Sub-6 GHz TNs.

Link failures can be detected by looking at different symptoms, our approach considers two: high number of retransmissions and number of missed beacons. The former would detect a broken link when transmissions to a given peer require a high number of retries (e.g. exceeding a given threshold). This method provides a fast detection but, on the other hand, it requires active communication between the peers. This means that it will only detect a link failure if there is an active flow using that link; otherwise the broken link will go unnoticed until it is actually needed. Note that a path reconfiguration at that moment will introduce delays that could have been avoided with an earlier detection. This earlier detection could be achieved by implementing the second method, based on counting missed beacon frames.

In the proposed 5G-XHaul architecture [1], wireless TNs form a multi-hop wireless mesh backhaul network. IEEE 802.11-based Sub-6 GHz TNs use the mechanisms defined by the IEEE 802.11s amendment [4], which uses periodic beacon frames to discover and maintain the topology of the mesh. Failure to receive such frames is indicative of a broken link and can thus be used to trigger any failure recovery protocol in the node. Mesh nodes can miss beacon frames due to the following reasons:

a. The beacon was received with errors due to temporary bad channel conditions (error-prone wireless channel).
b. The beacon was received with errors due to a persistent bad channel or heavy interference.
c. The beacon was not sent due to malfunction of the peer node.
d. The beacon was not sent because the transmitter did not find the channel free (indicative of severe interference problem).

Under normal operation (i.e. good link conditions), nodes would seldom miss a beacon although that is still possible due to the nature of the wireless channel. In that case (a), activating failure recovery mechanisms can be counterproductive. However, missing multiple consecutive beacons would imply, with high probability, that the link is no longer usable (cases b, c, d). How many missed beacons are enough to determine a broken link and how often those beacon frames should be sent pose a new trade-off: frequent beacons will allow a rapid detection of a broken link at the cost of increasing signalling overhead.

Finally, also note that, while detection based on missed beacons allows broken link detection (link completely unusable) even with no ongoing data transmissions, the retransmission-based detection allows the discovery of any “lossy” links. The level of admissible losses in a link not to be considered broken is configurable through a retransmission count threshold.

3.2 TDMA slot time configuration

Recall that Sub-6 GHz BH TNs will probably compete for the same (scarce) radio resources used by the radio access network (RAN) and, therefore, the services of an intelligent radio resource management become a requirement. We argue that a centralized scheduling of access and backhaul links using hybrid TDMA/CSMA access scheme will allow to fairly distribute resources between both network domains and to reduce collisions, which are inherent to Wi-Fi communications. What is more, the use of TDMA would enable the application of advanced services such as network slicing. For this novel application, we use hMAC [5], a technique proposed by the TKN group at the TU Berlin to provide a TDMA-like access in Wi-Fi interfaces. The hMAC implementation is based on the idea of using a hybrid TDMA/CSMA medium access control protocol, where local or remote (i.e. SDN controller) scheduling of the different IEEE 802.11’s Enhanced Distributed Channel Access (EDCA) queues at the driver level, controls outgoing flows. Later, the “real” over-the-air transmission is achieved by following the default IEEE802.11 CSMA/CA rules.

In this architecture, the controller manages time slot configuration at each radio link. The controller can apply any scheduling algorithm based on the number of flows going through each link, the weight of each flow, the radio characteristics of the link, etc. or any combination of such parameters (reported via OpenFlow as explained before). The hMAC controller then communicates the scheduling to the access network nodes (*ANx*) and BH transport nodes (*TNy*) using an agent deployed at each device. Figure 3-1 provides an example of such hybrid TDMA scheduling, where four TN nodes and two AN nodes share two radio channels (Ch. *X* and Ch. *Y*) to serve three data flows.

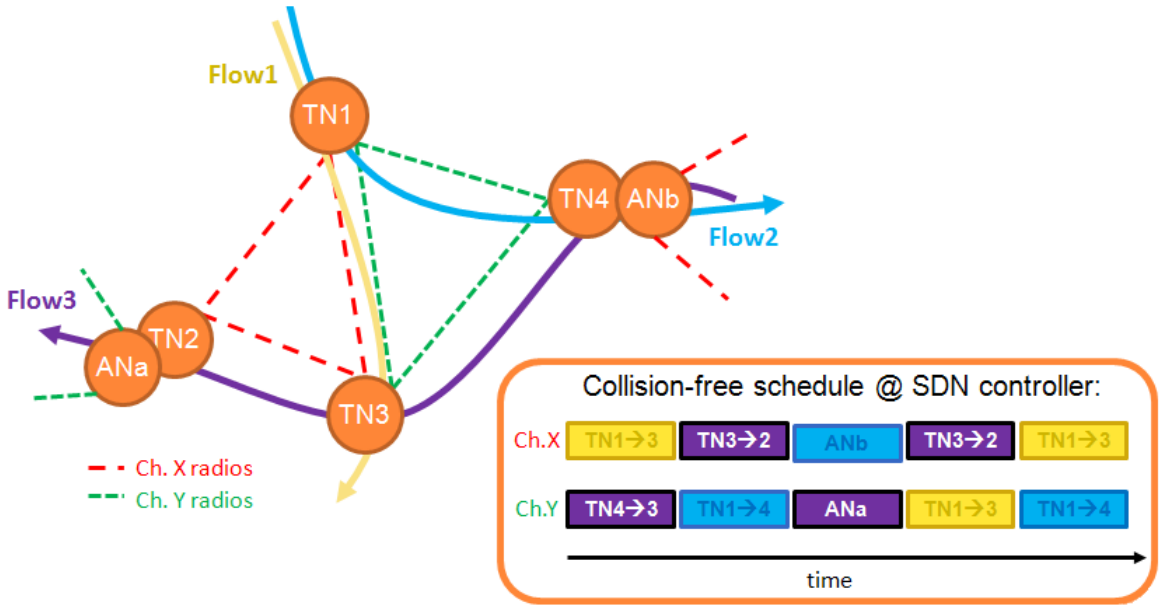


Figure 3-1: Example of scheduling of a TDMA-based backhaul and access network infrastructure.

Each hMAC-enabled AN/TN allows the configuration of its EDCA queues exposing the following parameters to enable the hybrid TDMA/CSMA access:

- T_s: duration of a single slot
- N_s: number of slots per TDMA frame
- ***Mapping***: descriptors of the traffic allowed on each slot
- T_0: absolute time at which the first slot is started.

4 Evaluating the applicability of the proposed management capabilities of Sub-6 GHz TNs

In this section, we provide further details of the implementation of the Sub-6 GHz TN's features defined in sections 2 and 3, and evaluate the feasibility of different applications of such features as a proof of concept: topology construction, interference-aware routing and CPU-aware routing enabled by network state monitoring capabilities, link failure detection and TDMA slot configuration.

4.1 Topology discovery

As mentioned in section 2.2.2, the first problem to solve when the Sub-6 GHz BH is deployed is the construction of the signalling path; that is, TNs must find a path towards the controller (not necessarily an optimal path). In a first approach, SDN agent ***A*** populates the OpenFlow tables according to a pre-established tree-like topology where one of the OpenFlow ports is designated as the *parent* port (port used to deliver packets addressed to the controller), and the remaining ports are designed as *child* ports (used to relay controller signalling to neighbouring wireless TNs in the network). This behaviour requires i) that the selection of *parent* and *child* interfaces is defined offline (e.g. following a shortest path criterion), and ii) to initialize the wireless switches sequentially, starting from the gateways - i.e. switches with direct (wired) connectivity to the controller. We have to note that this process is required only once at installation time. Once the initial connection has been established, the actual path followed by the signalling connection can be optimized from the controller.

Using OpenDayLight [6], the view of the topology created in the example scenario of Figure 2-1 (assume TN1 acts as a gateway) is depicted below in Figure 4-1. Recall that, in order to build the topology, the TNs must be provided with an initial configuration (i.e. *parent* and *child* ports), in this case:

1. **TN1**: child (TN2, TN3)
2. **TN2**: parent (TN1); child (TN3, TN4)
3. **TN3**: parent (TN1); child (TN2, TN4)
4. **TN4**: parent (TN2); child (TN3)

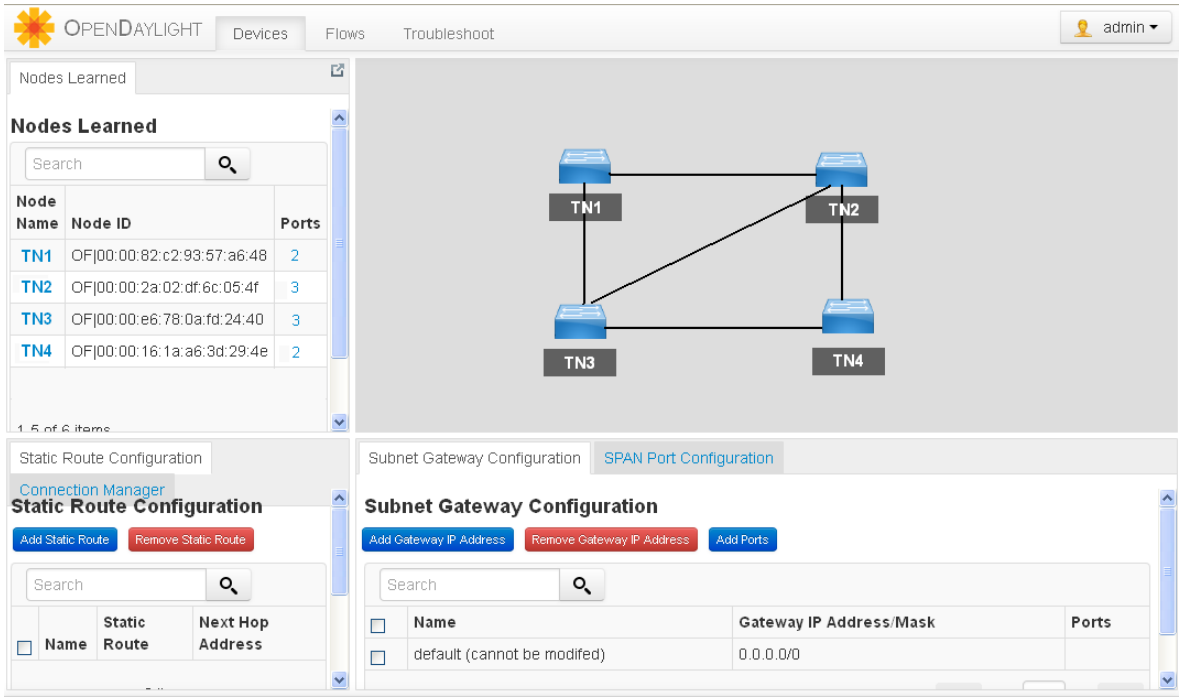


Figure 4-1: Controller (ODL) view of a Sub-6 GHz BH example topology.

We would also like to note that the abstraction of the broadcast nature of the wireless channel explained in section 2.1 does not hide the interactions between interfering links. Actually, the controller is able to build an

interference matrix ***I*** by determining which links are on the same channel in a node's neighbourhood[4] thanks to the interface statistics reported by the TNs (cf. section 3.1.1). Thus, the interference matrix corresponding to the five links in the example topology of Figure 2-1 would be:

$$I = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

From that moment onwards, an intelligent routing algorithm running in the controller (e.g. a Tier-0 controller [7]) can establish different paths for different data or signalling flows according to any desired metric by installing new forwarding rules on the managed wireless TNs.

4.2 Reporting of statistics

Once the topology is built, as explained in section 3.1 each of the nodes in the network maintains information about its internal state, its wireless network interfaces and ports, and the observed network environment. The SDN controller periodically polls each node for statistics (OpenFlow *StatsReq message*); nodes respond to such a request with statistic response packets (OpenFlow *StatsRes*) that contain all the statistics gathered over the last period.

Figure 4-2 shows a StatsRes packet extended to include the radio parameters described in 3.1.1.

The time between consecutive polls is configurable and poses an important trade-off. With frequent updates, the controller can react quickly to any event in the network but, on the other hand, it increases the signalling overhead, which scales with the number of managed TNs and the number of carried flows. Typical values in wired networks range from 20 to 30s but the dynamism of wireless channels would require lower values to guarantee that the controller processes up-to-date data. For example, updating those statistics every 16 s and assuming two or three peers (i.e. abstract switch ports) per TN, each polling period TNs send about 16kB of data in OpenFlow *StatsRes* messages to the controller, which translate to less than 8 kbps of signalling per managed Sub-6 GHz TN.

Receiving those series of measurements, the SDN controller calculates a moving average of the gathered values, stores them in a database, and processes the data in order to obtain a view of the network state. This network state is then used by the decision-taking algorithm to manage the network data plane, as exemplified in the following two subsections.

[4] In our current implementation, it is assumed that transmissions can interfere co-channel nodes up to two hops away.

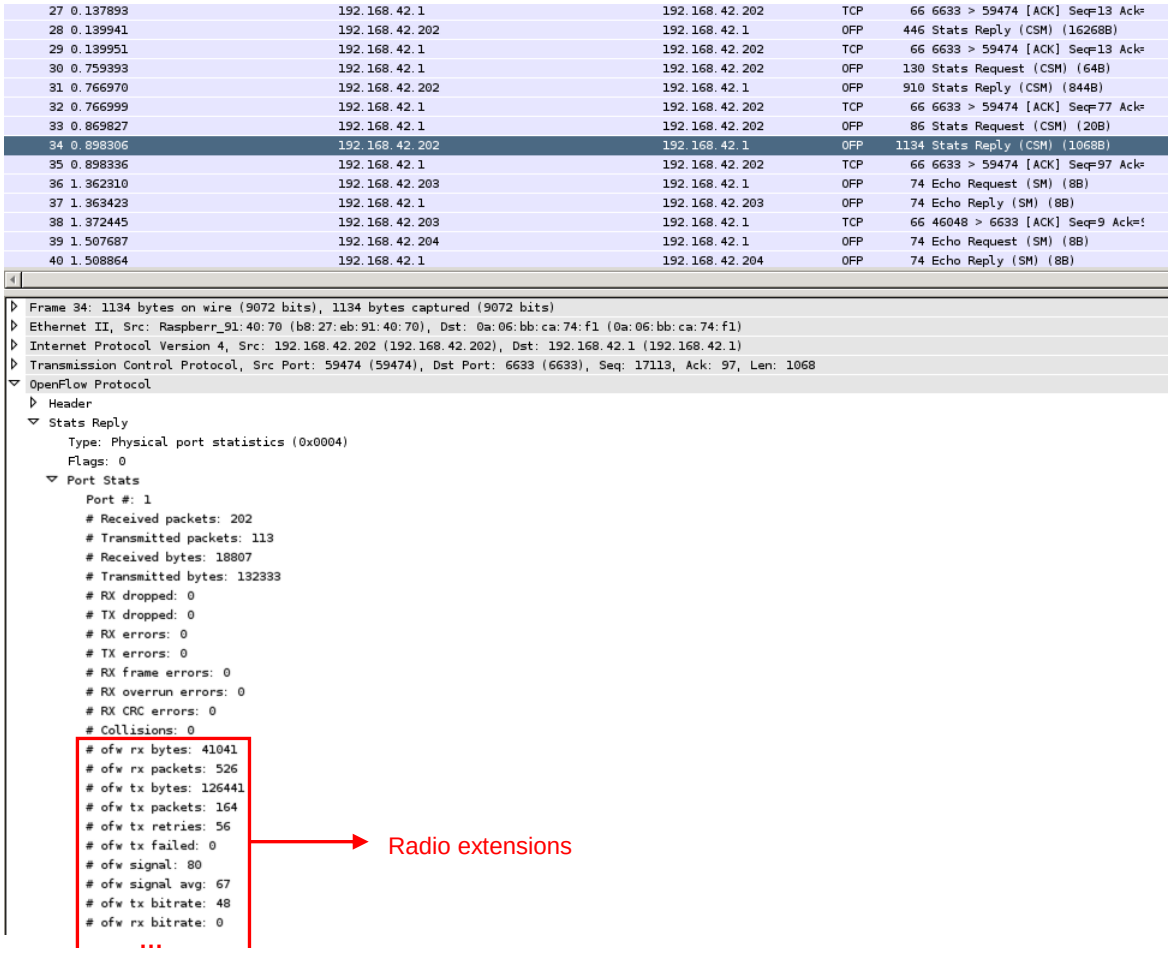


Figure 4-2: Captured OpenFlow StatsRes message with radio extensions.

4.2.1 Proof of concept 1: Interference-aware routing

Channel usage measurements are useful to detect congested channels and are therefore of utmost importance in order for the controller to be able to provide optimal data paths avoiding congested links.

Our wireless SDN implementation is based on open source components available in Linux platforms. Besides, in an attempt to maximize compatibility with a wider set of IEEE 802.11 devices, the SDN adaptation does not reach below the *mac80211*[5] module [8] thus remaining independent of the underlying wireless device driver. At that layer, channel utilization can be computed by measuring the time during which the radio has been receiving frames plus the time spent in transmissions. This would limit the sources of interference that can be detected to IEEE 802.11 transmitters. The portion of busy time due to reception/transmission is measured at fixed intervals (e.g. referenced to *Beacon Interval*) and averaged over several of such intervals.

This capability is demonstrated in the simple scenario of Figure 4-3, which depicts the wireless channels of the links set up between the nodes (Ch. 11 at crowded 2.4GHz band, and Ch. 48 at interference-free 5GHz), as well as an external interferer that operates on channel 48. Initially, the path from the source (***TN0***) to the traffic sink (***TN4***) uses the 5GHz links (i.e. TN0 → TN1 → TN2 → TN4).

[5] The *mac80211* is a Linux kernel module running the IEEE 802.11 MLME for all softmac wireless cards.

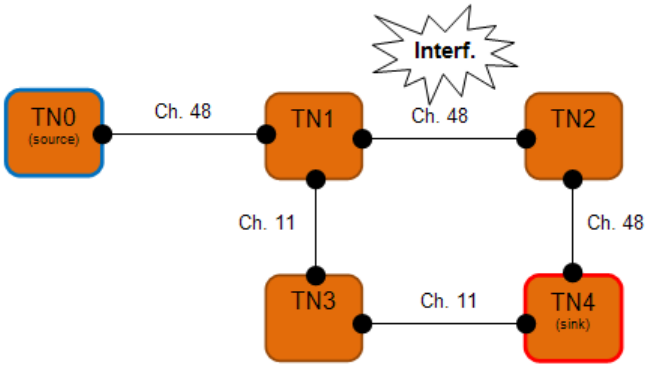


Figure 4-3: Example topology to test extended statistics reporting from Sub-6 GHz TNs.

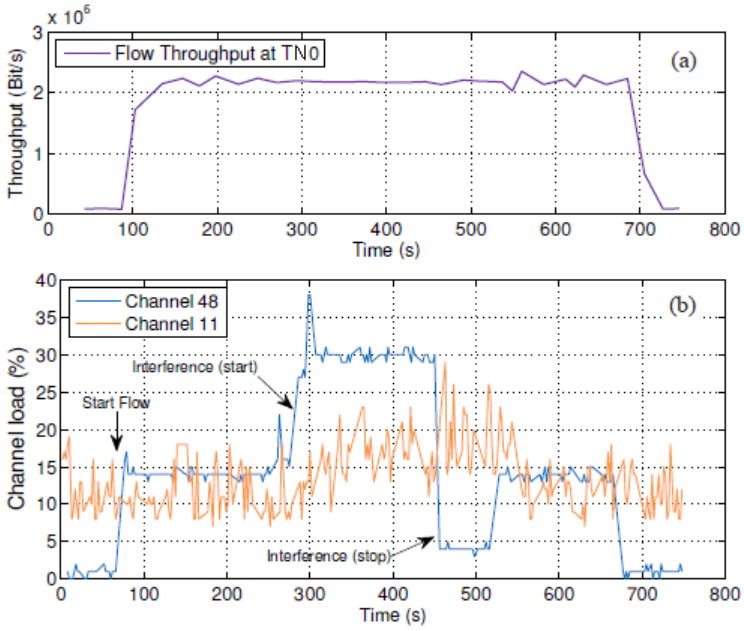


Figure 4-4: (a) throughput measured over link *TN0* → *TN1*; (b) channel utilization reported by *TN1* during interference-aware routing experiment.

Then, a flow from *TN0* to *TN4* with a data rate of 2 Mbps is generated. Figure 4-4 (a) shows the throughput statistics gathered for the *TN0*→*TN1* link at the controller. Figure 4-4 (b) displays the utilization measured on channels 11 and 48 reported by *TN1*, including the background traffic (~11% on Ch. 11) and the load due to signalling traffic (channel 48, ~1%).

Once the 2 Mbps data flow enters the network, the load of channel 48 increases to approximately 15% (*t*=80 s). Around 300 s into the experiment, the interferer starts transmitting on channel 48, clearly visible by a sudden peak of channel load. At this point the controller estimates a better network state through *TN3* and redirects the flow. After applying the reallocation, the load on channel 48 settles to a steady 30%. On the other hand, the channel load of channel 11 increases due to the flow reallocation. At *t*=450 s, the interferer stops transmitting and the load on channel 48 drops to around 5%. Then, the controller decides to balance the network load again: shortly after the 500 s mark, the algorithm switches the data flow back to channel 48.

4.2.2 Proof of concept 2: CPU-aware routing

In edge computing scenarios where, for example, TNs may have virtualization capabilities and hold VNFs of different tenants, those edge nodes may become temporarily CPU hogged due to the execution of local computations, which affects their data forwarding capabilities. Thus, awareness of CPU load of Sub-6 GHz backhaul nodes will allow the controller to minimize the participation of those nodes undergoing high CPU utilization in loaded data paths. In this experiment, a CPU-aware routing policy is evaluated, using the setup depicted in Figure 4-3, changing now all links to channel 11. A 1 Mbit/s flow is created at *t*=40 s, using *TN3* as relay node

between the source and the traffic sink, as can be derived from Figure 4-5 (a). At approximately *t*=240 s, a CPU-intensive task is started in *TN3*, boosting the CPU load to 100%, as shown in Figure 4-5 (b). The SDN controller reacts to the increased CPU load by redirecting the flow over *TN2*, thus alleviating *TN3* from data forwarding tasks and granting it more processing power to perform the computing task.

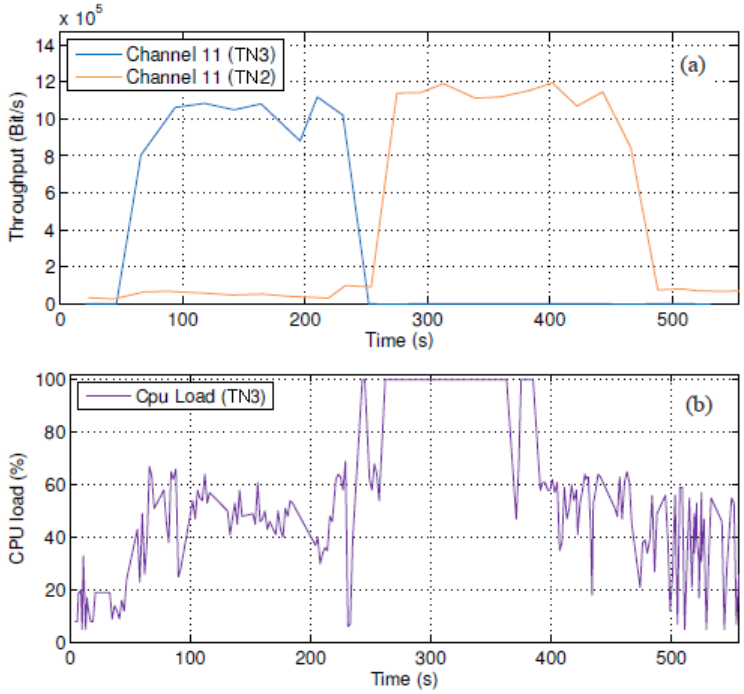


Figure 4-5: (a) throughput measured at TN1; (b) CPU load reported by TN3 during CPU-aware routing experiment.

4.2.3 Detection of broken links

As mentioned in section 3.1.2, the beacon interval (*BI*) or the number of missed beacons to wait before determining a broken link impact the performance of path recovery mechanisms.

As a first approach, we modify *mac80211* kernel module (cf. 4.2.1) to notify a user-space application after N_{MB} missed beacons. The user-space application would then initiate the failure recovery procedure (e.g. activate a back-up path and immediately notify the controller).

Figure 4-6 shows the maximum reaction time achieved with different values for the beacon interval, expressed in time units (TU) of 1024 µs when the node is configured with N_{MB} = 3. The maximum reaction time assumes the worst detection time, when the link is broken immediately after a beacon transmission; in that case, a link failure cannot be detected before N_{MB} x *BI* plus the time required to process the notification. More precisely, the maximum reaction time is the sum of the following components:

- **Kernel time**: time between the last beacon successfully received and the moment the kernel generates the alarm. This time includes N_{MB} x *BI* plus kernel processing delays.
- **Notify time**: time between the end of the kernel time and the moment the alarm is logged in the filesystem (*kernel.log*) and notified to the user-space script through *inotify*[6] API.
- **Script time**: time used by the user-space *bashscript* application parsing the log and the failure recovery procedure is initiated.

[6] *inotify* is a Linux kernel subsystem that notifies applications of changes in the filesystem.

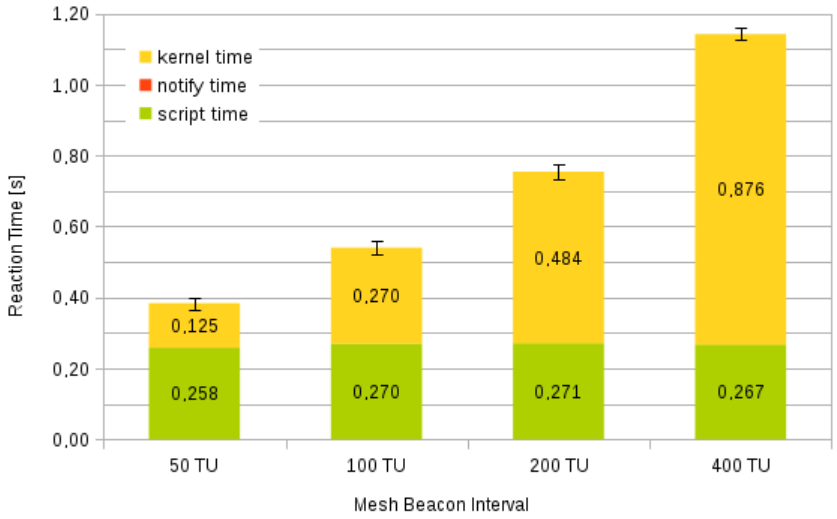


Figure 4-6: maximum reaction time after a link failure detected with two missed beacons.

As expected, the kernel time scales with *BI*, while the rest of the contributions remain constant. Notify time is negligible, but the user-space application's processing time is of the same order of magnitude as the kernel time. We think that a more efficient implementation of the notification processing (e.g. through a better communication with kernel-space and a lighter programming language for the user-space application) will considerably reduce the script time.

4.3 Hybrid CSMA/TDMA operation

The scarce radio resources available under 6GHz shared between RAN and BH require an intelligent management. Centralized scheduling of access and backhaul links using hybrid TDMA/CSMA access is an appealing solution to fairly distribute resources between both networks and shows the added advantage of enabling network slicing to accommodate different services or even different operators on the same infrastructure. In this section, we explore the possibilities that TDMA-enabled Sub-6 GHz nodes can offer to 5G-XHaul control plane.

4.3.1 Synchronization

Note that, with the specification of parameter T_0 (section 3.2), the deployment of a dynamic scheduling by the controller requires a precise synchronization of the nodes. In our experiments, synchronization is provided through a common wired infrastructure. However, in a real deployment, such infrastructure will not be present and synchronization is provided over the air. This poses other interesting challenges under discussion in 5G-XHaul's WP4. Synchronization based on IEEE 1588 PTP (Precision Time Protocol) showed 99.0% of samples below the millisecond accuracy level, and up to 91.1% of them below 100µs, being the average of the time difference about 50µs. Therefore, we can conclude that PTP provides enough precision for the time scales of the optimized hMAC, where slot durations range from 1 to 20ms (as explained in following sections). Figure 4-7 depicts the results of a 2-hour PTP experiments.

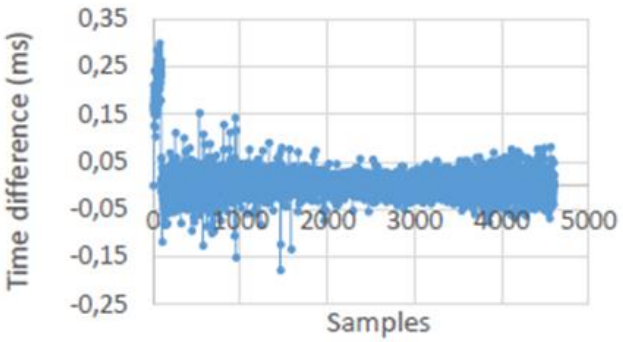


Figure 4-7: Time difference between two nodes synchronized with PTP.

4.3.2 Fine tuning of TDMA slots

One of the most interesting features of the proposed Sub-6 GHz TNs is the new hybrid MAC layer that combines the scheduling of TDMA (found in other cellular deployments such as GSM) over legacy IEEE 802.11 CSMA/CA scheme; hMAC (Hybrid CSMA/TDMA MAC) [5] implements such kind of mechanism. The general idea behind its implementation is to work with software packet queues that take care of opening and closing a set of temporal slots into which the working cycle is divided. Each slot can be defined with different QoS and addressing rules for the filtering of outgoing flows. However, under this software queues, legacy IEEE 802.11's distributed coordination function (DCF) keeps working with its usual CSMA-like operation so that the transmissions actually happen with this classical listen-before-talk protocol.

After a large set of experiments run to obtain the optimal values for each of the configurable TDMA parameters, we derived the following conclusions:

- T_s should have a duration of, at least, ten times the transmission time (T_m) of a single packet (including interframe spaces and 802.11's ACK transmission).
- The shorter the time slot (T_s), the better the performance both in terms of jitter and throughput.
- The lesser the number of slots (N_s), the better the performance in terms of both jitter and throughput.

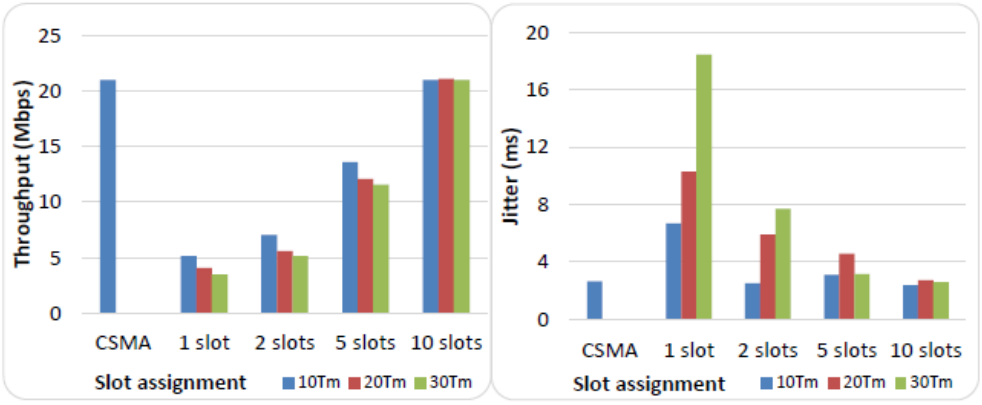
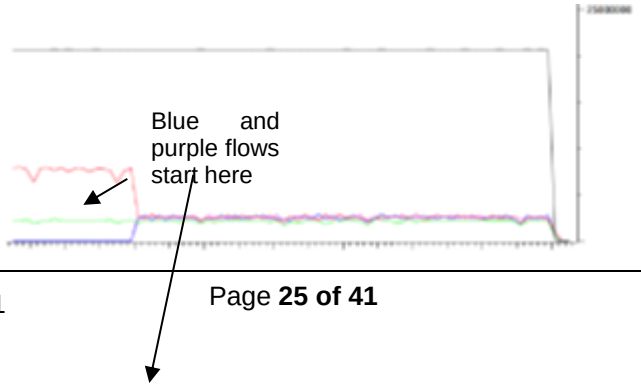


Figure 4-8: Example results of throughput and jitter measurements with a CSMA and hMAC approach where 1, 2, 5 and 10 out of 10 slots are open, and using different slot duration (in terms of frame transmission time T_m).

hMAC can also be a useful tool for Quality of Service (QoS) provisioning in shared wireless medium, as it enables more precise resource management techniques with a wider variety of options to dynamically accommodate the medium to the requirements of different transport classes (TC) defined in [1] or, due to the limited capacity of Sub-6 GHz links, to apply different policies to different subclasses within the lowest TCs. Legacy CSMA relies on EDCA for the provisioning of QoS, which provides random access-based traffic differentiation using four different categories. Although it uses the same number of queues, hMAC adds time as a new dimension, which enables an increased flexibility and eases a more precise QoS provisioning. Figure 4-9 shows an example of such flexibility, where CSMA/CA offers fairness among categories, and hMAC is able to achieve max-min fairness on a per-flow basis. Figure 4-9 illustrates the throughput of five flows (**black** line: high priority; **red**, **blue** and **purple** lines: medium priority; **green** line: low priority). The addition of two new medium-priority flows (blue and purple) reduces the QoS within that category in EDCA, but hMAC allows any new distribution; for example, redistributing resources among the different categories in order to keep fairness levels.



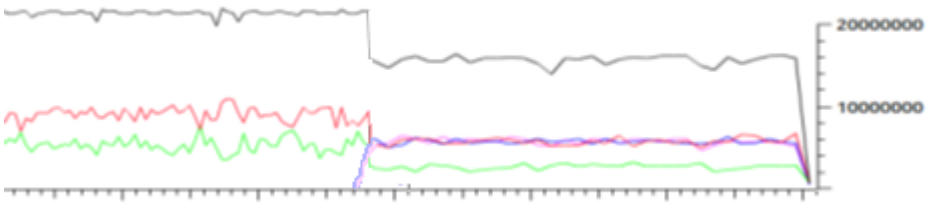


Figure 4-9: QoS fairness study with 5 simultaneous flows using CSMA (top) and hMAC (bottom).

Finally, after tuning hMAC parameters in a single link scenario, we tested it in a multi-transmitter scenario where three stations compete (stations ***A***, ***B*** and ***C***) in saturation conditions (traffic generated by *iperf:* 1000B UDP datagrams, generating 10Mbps per flow). Figure 4-10 shows the results of those experiments, in four different configurations:

- CSMA: shared medium competition based on legacy IEEE 802.11's CSMA/CA.
- HTDMA: competition based on hMAC slot assignment as *AAAABBBBCCCC*, i.e. time frames of 12 slots of equal duration, with 4 consecutive slots for each node.
- HTDMA alternate: competition based on hMAC slot assignment as *ABCABCABCABC*, i.e. the same 12 slots but here with an alternate assignment.
- HTDMA shared: competition based on hMAC, with a slot assignment equivalent to *XXXXXXXXCCCC*, wherein X slots are shared by ***A*** and ***B*** in the standard CSMA/CA fashion, while ***C*** are slots reserved for the third node.

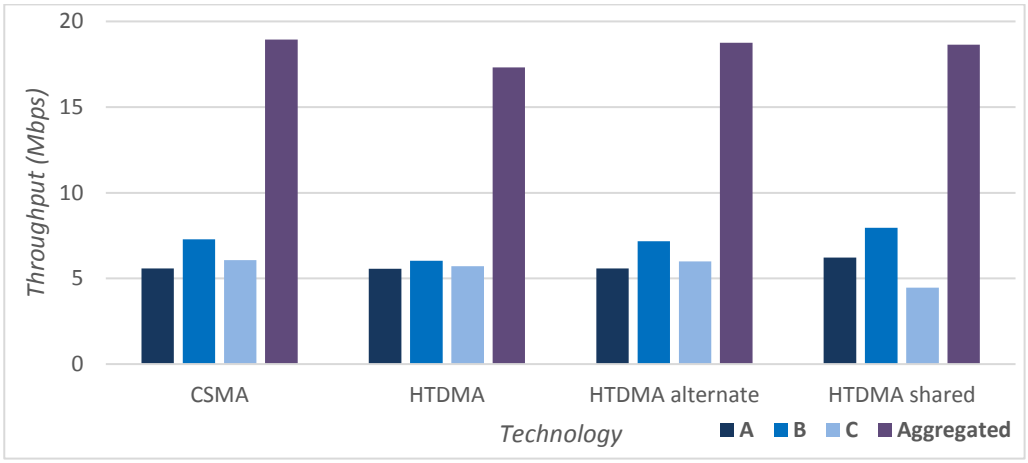


Figure 4-10: Throughput measurements with different configurations in a competition environment.

Even though all three stations compete in the same conditions, random access results in unfair resource sharing, in favour of station ***B***, in this case. It is possible that ***B*** has skewed tolerances in the CSMA slot time and tends to discount backoff timer faster than the others (even if the same CW is used). We argue that enabling hybrid TDMA solutions can minimize the impact on fairness of this endemic problem of CSMA. As depicted in Figure 4-10, hMAC provides a fairer resource sharing, as the access to the medium is not based on random attempts but on predefined slots. However, the aggregated throughput seems to be slightly reduced. Moreover, from this figure we can also deduce that faster flow switching or slot sharing can produce CSMA-like behaviour as a result of the implementation of the hMAC mechanisms.

5 Data Plane performance of Sub-6 GHz devices

The initial evaluation of 5G-XHaul Sub-6 GHz TNs' programmable capabilities presented in the previous sections was carried out in different testbeds with different hardware and software platforms – Raspberry Pi, laptop and desktop computers. However, in order to ease the integration of 5G-XHaul TNs described in this document, the architecture presented in section 2 is finally implemented in a definite platform. In this section, we introduce the 5G-XHaul's Sub-6 GHz Access and Backhaul wireless nodes that will be used henceforth throughout the project. We first introduce the prototype boards, and then describe a preliminary evaluation carried out in an indoor scenario to validate the achieved data-plane performance of the device. The tests described in this section will be extended during the next year in the framework of Work Package 5 (WP5) with outdoor testing, before taking these devices to the integrated 5G-XHaul testbed in **BIO**.

5.1 Physical description of the Sub-6 5G-XHaul devices

Figure 5-1 depicts the prototype 5G-XHaul Sub-6 GHz wireless device. The device is based on a single board computer (SBC) provided by Gateworks (Ventana GW5410[7]), featuring a Quadcore ARM-Cortex A9 processor, 1 GB of RAM and 2 Gb Ethernet ports. The board is equipped with six mini-PCIe ports to connect network interface cards (NICs). In our case, the board features three wireless NICs manufactured by Qualcomm for the consumer market. Two of those wireless NICs are based on the QCA9880 chipset featuring IEEE 802.11ac with 3x3 MIMO, hence operating only in the 5GHz band. One of the 802.11ac NICs is placed on top of the board, as depicted in the left hand side of Figure 5-1, and the other at the back of the board, as depicted in the right figure. In addition, another wireless NIC, based on the AR9280 chipset featuring IEEE 802.11n with 2x2 MIMO, is placed on the back of the board (see right hand side of Figure 5-1). Notice that 802.11n is the fastest commercially available IEEE 802.11 technology operating in the 2.4 GHz band. In the described wireless device, the two IEEE 802.11ac NICs operating at the 5GHz band will be used for backhauling purposes, and the 802.11n interface operating at the 2.4GHz will be used for access. However, the 802.11n NIC is also capable to operate also in the 5GHz band and, therefore, access and backhaul may compete for the same spectrum.

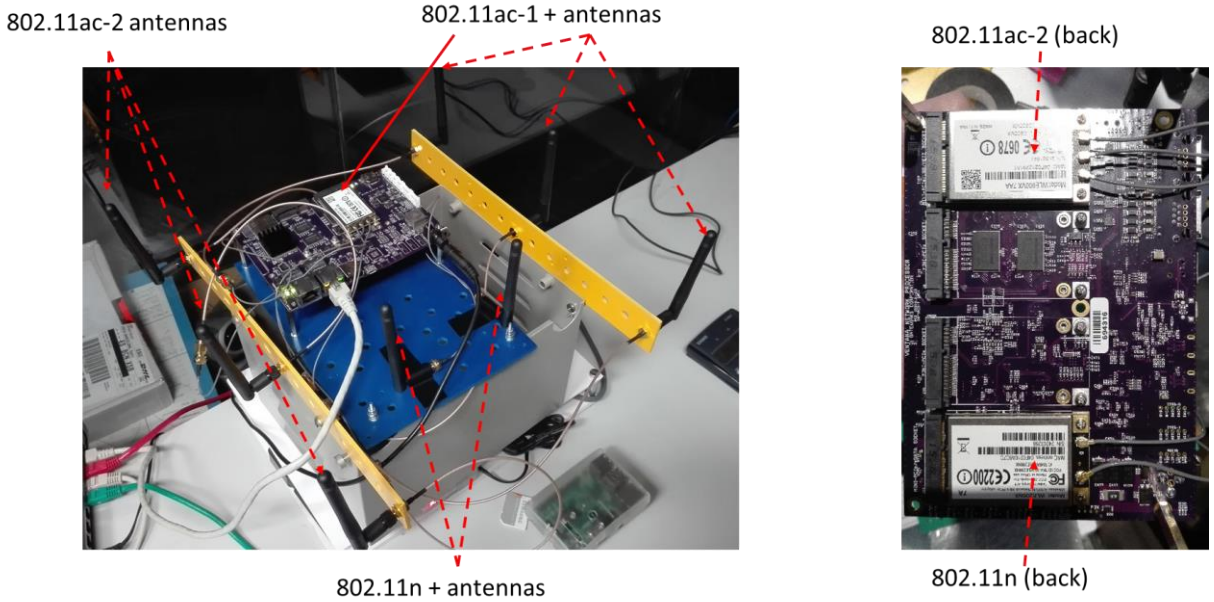


Figure 5-1: 5G-XHaul Sub-6 wireless development board.

The design principle adopted for the 5G-XHaul Sub-6 GHz wireless device is to reuse, as much as possible, technologies designed for the consumer market, hence benefitting from low costs due to economies of scale, and then try to build in software carrier grade features, as detailed in sections 2 and 3.

[7] http://www.gateworks.com/product/item/ventana-gw5410-network-processor

5.2 Initial measurement campaign of the 5G-XHaul Sub-6 wireless device

In this section, we report the results of an initial indoor measurement campaign performed to characterize the performance of the 5G-XHaul Sub-6 GHz device, and to optimize its configuration. Subsequent measurement campaigns will follow, as part of WP5.

The measurements reported in this section were carried out at the I2CAT premises, after office hours (later than 7pm) in order to avoid most of the interference in both the 2.4 and 5GHz bands; note, however, we cannot guaranteed that channels are completely interference-free. Two 5G-XHaul Sub-6 GHz devices are positioned as indicated by the red dots in Figure 5-2, at a distance of 10 meters and NLoS conditions. Performance is then tested by launching a data stream that saturates the channel using the *iperf* tool. As described in D2.2 [1] higher distances are expected in dense Small Cell deployments, however, we choose to perform this first tests using a smaller distance to understand the maximum performance provided by the Sub-6 GHz node. Subsequent tests will be performed in an outdoor environment with longer distances.

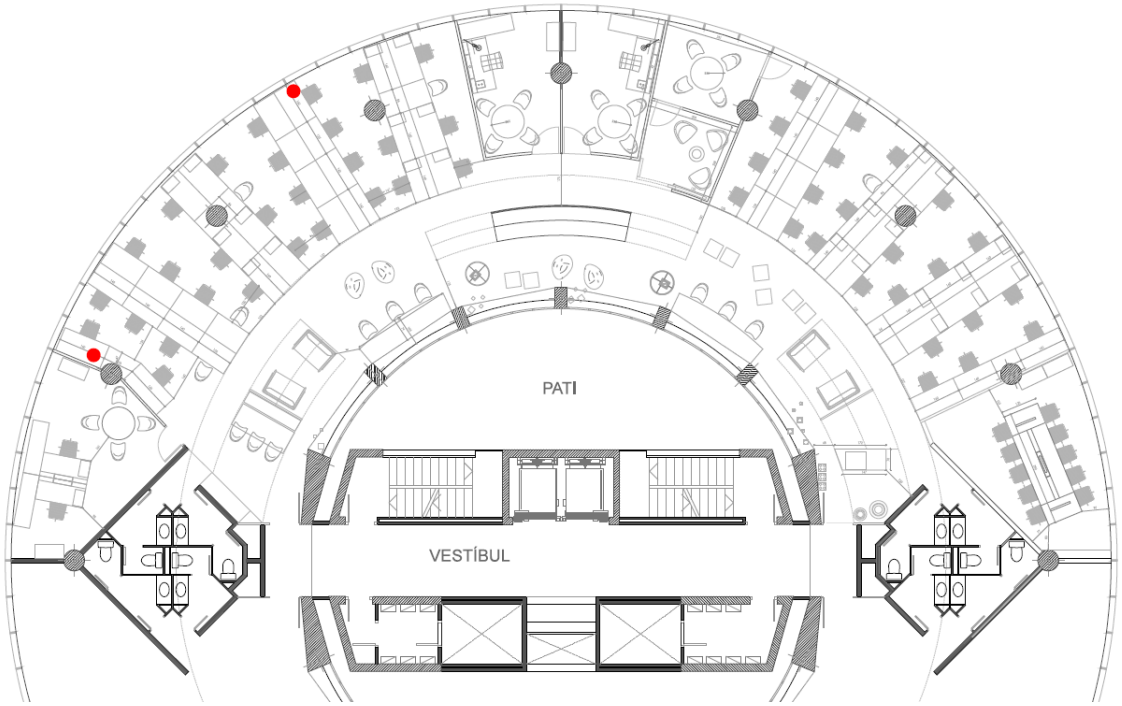


Figure 5-2: Measurement set up.

In order to maximize the bandwidth available in the wireless backhaul, the two IEEE 802.11ac NICs are configured to operate in 80MHz mode. In addition, higher channel bandwidths spread the available TX power over the whole bandwidth and have thus a reduced range [9], which is considered beneficial for dense Small Cells deployments. Figure 5-3 describes the 802.11ac channel configuration in different regulatory domains.

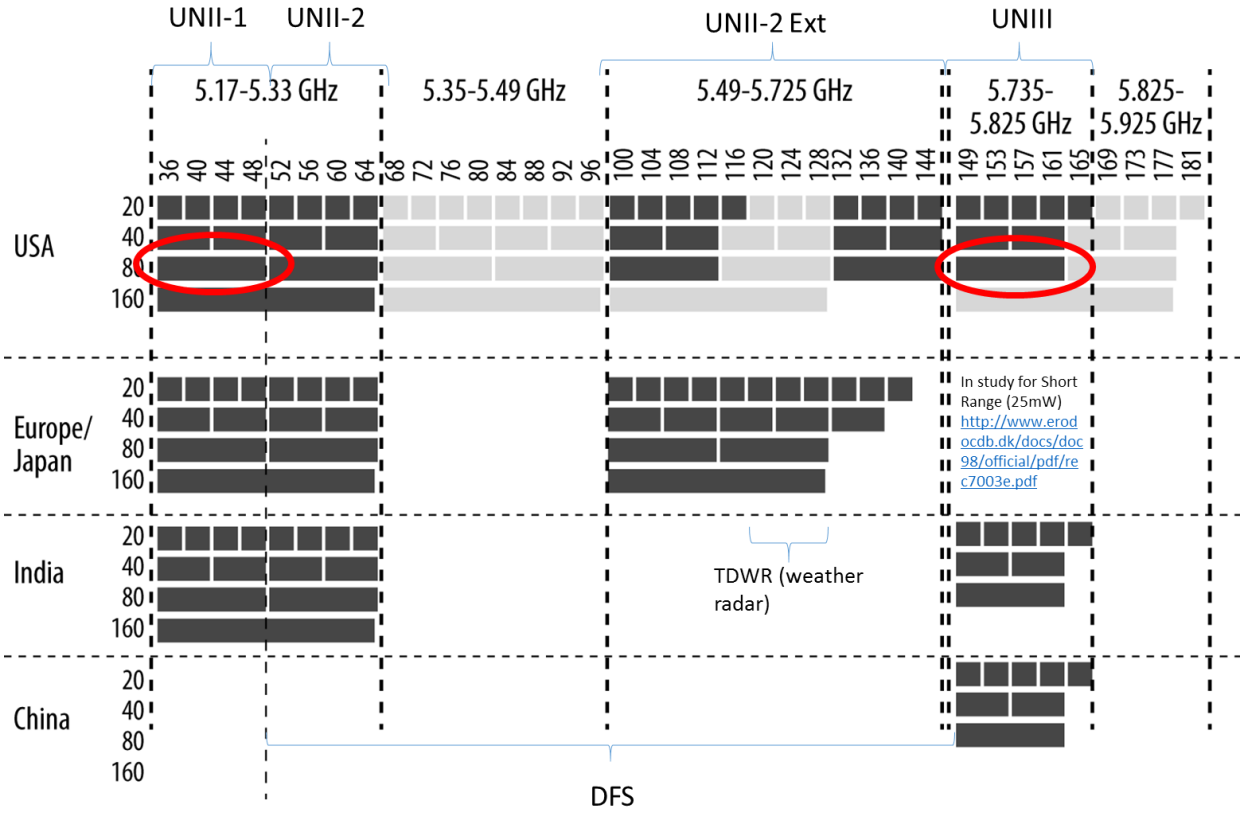


Figure 5-3: Worldwide channel allocation for 802.11ac

For the purpose of this study, we look at the US and Europe/Japan regulations, and assume the highest number of available channels when combining both domains. This assumption is in line with current discussions on the use of the 5 GHz ISM band for WLAN, which should result in new rulings on WRC-19 (e.g. see [10]). An important constraint when operating in the 5GHz band is that some channels can only be used if Dynamic Frequency Selection (DFS) is supported in the wireless NICs. The reason for DFS is to protect weather radars operating in the 5GHz band from WLAN transmissions. Both Europe and the US require DFS between channels 68 and 144 (c.f. Figure 5-3). However, in order to maintain the cost of wireless equipment low and to reuse technologies used in the consumer space, we avoid the use of DFS. Therefore, in our study we consider only 36 and 149, highlighted with a red circle in Figure 5-3, as the only 80 MHz channels available for wireless backhauling. The use of 160 MHz channels, although defined in the IEEE 802.11ac standard, is not yet available in the first wave of commercial NICs, and hence not supported by our board. Recall that this is only a convenient implementation of 5G-XHaul Sub-6 GHz nodes for testing purposes, in a commercial deployment of the 5G-XHaul architecture, other bands (including licensed and license-free) and channelization can be considered.

Figure 5-4 describes the test methodology followed to characterize the performance of the 5G-XHaul Sub-6 GHz wireless device. We start characterizing the performance of NICs operating in isolation, since this is the set up where we expect the maximum throughput. We continue analysing the performance of the device when the two backhaul IEEE 802.11ac NICs operate concurrently in channels 36 and 149. Although these two channels are far from each other in the 5GHz band, based on previous works (e.g. [11] for IEEE 802.11a) we expect a certain degree of cross-card interference. In addition, the dual NIC setup is tested in two configurations, a *direct* one, where the two cards transmit from one peer to another, which could be seen as a case of link aggregation, and a *crossed* configuration where one card receives while the other transmits, which can be seen as a the device acting as relay inside a mesh. Finally, the triple NIC setup is studied, also in *direct* and *crossed* modes, in order to understand if there is interference between the 802.11n NICs operating in the 2.4GHz and the 802.11ac NICs operating in the 5GHz band.

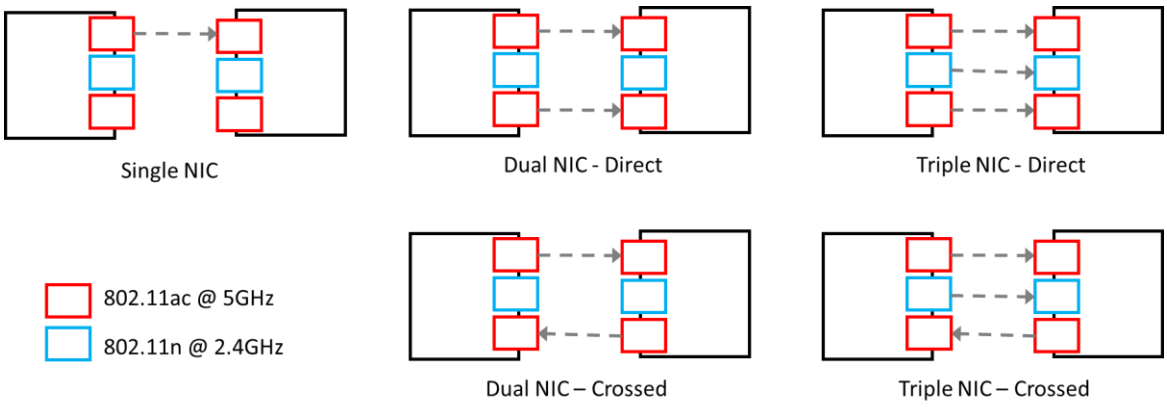


Figure 5-4. Test methodology.

5.2.1 Single NIC experiments

Figure 5-5 depicts the results of the single NIC experiments, namely the instantaneous application-level throughput (*iperf* UDP), as measured by the receiver card. Figure 5-5 depicts a set of lines corresponding to different sets of experiments performed separately. Three types of experiments are performed:

- An 802.11ac TX/RX pair configured as mesh points (MP), and communicating through channels 36 and 149 (c.f. Figure 5-3).
- An 802.11ac TX/RX pair configured as Access Point (AP) and STA, and communicating through channels 36 and 149.
- An 802.11n TX/RX pair configures as AP and STA, and communicating through channel 4 in the 2.4GHz band but employing 40MHz channel bandwidth.

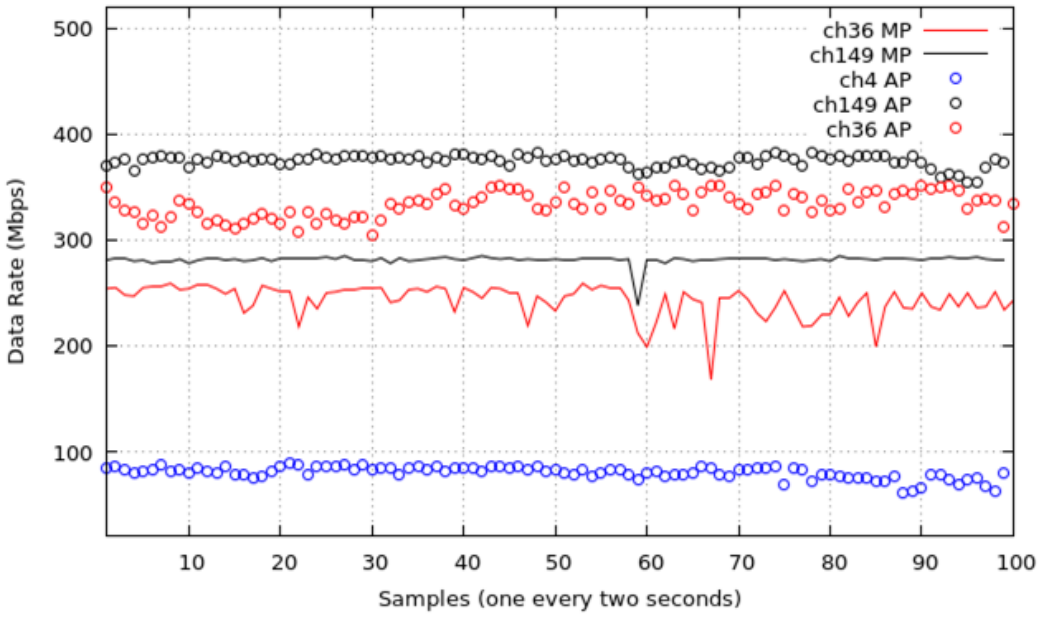


Figure 5-5: Single NIC experiments (tests are not performed concurrently).

Looking at Figure 5-5, we extract the following conclusions:

- AP/STA delivers higher throughput than MP (mesh mode) operation both in channels 36 and 149. We have to note that all our transmitters are configured to use rate adaptation using the Minstrel algorithms available in the Linux kernel [12]. Analyzing in detail the performed experiment we can see that, when configuring in AP mode, the 802.11ac NICs alternate between MCS 7 (64QAM 5/6)

with 3 spatial streams, and MCS 9 (256QAM) with 2 spatial streams (max PHY data rate of 975 Mbps). However, when configured in MP mode the 802.11ac only use a single spatial stream, although achieving MCS 8 (256QAM 3/4) and MCS 9 (256QAM 5/6) (max PHY data rate of 433 Mbps). There is no apparent reason why MIMO should not be available when configuring our cards in MP mode, and so we attribute this behavior to a software bug, which we will continue to investigate. This is a relevant problem because the software architecture described in section 2 requires the wireless NICs to be configured as mesh points. In addition, the observed application throughput is significantly below the data rate delivered by the PHY, especially in AP mode when 3 spatial streams and 64QAM are used, which results in a PHY rate of 975 Mbps. To investigate the reasons for this low MAC efficiency the upper part of Figure 5-6 depicts a sample of the instantaneous MAC aggregate size[8] (A-MPDU size) used by the device in AP mode and in MP mode. We can see that in AP mode a stable A-MPDU size of around 10 frames is used, whereas in MP mode the used A-MPDU size alternates between high and small values, with an average of around 20 frames. The higher average A-MPDU size in MP mode explains the higher MAC efficiency. More investigation is required to fully map the aggregation behavior of the wireless NIC, which is important for an SDN controller to discover the available capacity of a given network link.

- Channel 149 consistently delivers higher performance than channel 36. This behavior is expected because channel 36 and channel 48 are used by some networks visible from our office, which may interfere with our experiment. These interfering networks operate in 20MHz mode, thus reducing the transmission opportunities of our 80MHz links, which need to wait until both channels 36 and 48 are free before transmitting. 802.11ac supports dynamic bandwidth that could alleviate this degradation by downgrading the used bandwidth to 40MHz when channel 48 is busy. In WP5 we will study the advantages provided by this mechanism [9].
- The IEEE 802.11n-based RAN interface achieves a throughput around 80Mbps, as expected for an IEEE 802.11n link operating with 40MHz of bandwidth. We also have to note that this channel is subject to interference from overlapping Wi-Fi networks operating in the 2.4 GHz band.

[8] In 802.11ac the MAC aggregates a set of frames coming from the upper layer (MPDUs) into a single MAC frame (A-MPDU), which reduces channel access overhead hence increasing efficiency

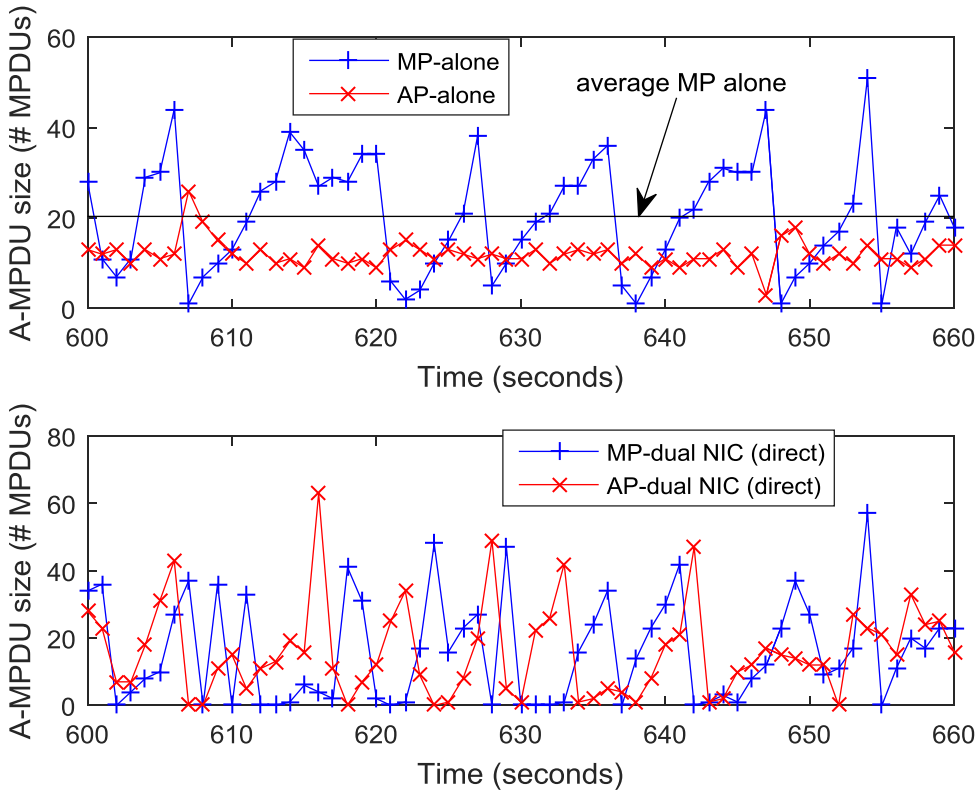


Figure 5-6. Size of MAC aggregates in AP and MP modes. Single NIC in upper graph. Dual NIC in lower graph.

5.2.2 Dual NIC experiments

The goal of the dual NIC experiments (c.f. Figure 5-4) is to asses if the two 802.11ac NICs operating simultaneously interfere each other even though they use distant channels (ch. 36 and 149). Previous works that analyzed multi-radio devices using legacy IEEE 802.11 radios, observed interference even when wireless NICs where configured in opposite extremes of the band [12]. We intend to validate if this situation aggravates with the use of 802.11ac and wider channel bandwidths.

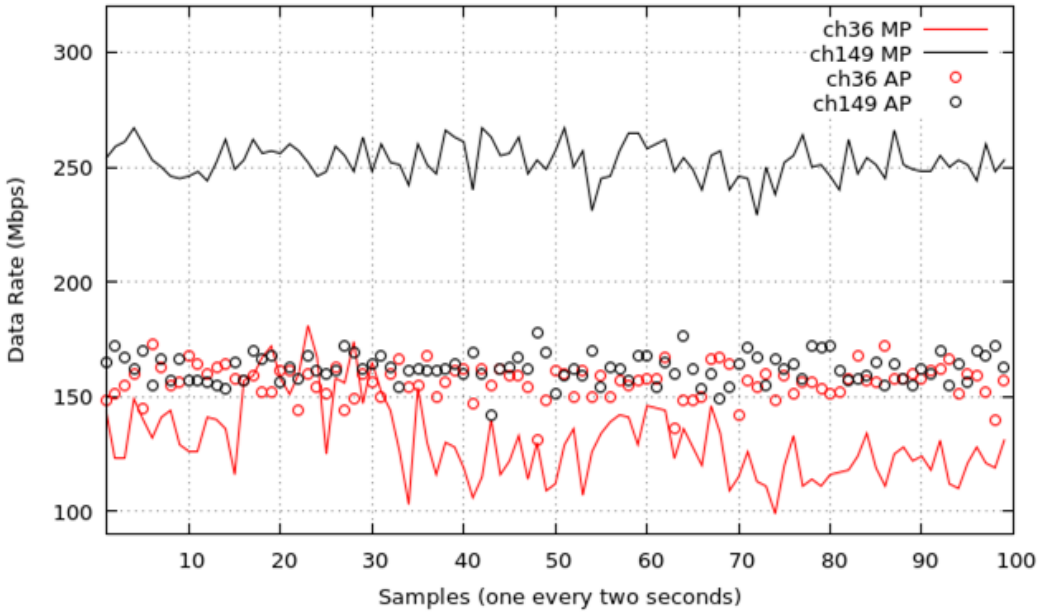


Figure 5-7. Dual NIC experiment in direct mode.

Figure 5-7 depicts the results obtained for the direct mode configuration (c.f. Figure 5-4), when the transceiver pair is configured in AP/STA mode or in MP mode. The figure shows the throughput obtained by each individual stream. The main findings from this experiment are:

- Tested direct links undergo significant cross-NIC interference. In AP/STA mode an aggregate throughput of around 320 Mbps, which is a very significant reduction from the 720 Mbps theoretical aggregate throughput obtained by combining the performance of channel 36 and 149 measured individually in the single link experiment (c.f. Figure 5-5). Analyzing the AP/STA transmission we can see that, in this case, the AP/STA transceiver pairs transmit a mix of MCS 5 and 3 spatial streams (PHY rate of 702 Mbps) and MCS 7 with 2 spatial streams (PHY rate of 650 Mbps), which is below the PHY data rates observed in the single NIC experiment. In addition, retry rates[9] of 19% in channel 149 and almost 25% in channel 36 are observed in the dual NIC experiment, whereas retry rates below 1% were observed in the single NIC experiment.
- The MP configuration results in an aggregate throughput around 375 Mbps, now slightly higher than the AP mode, but still well below the 520 Mbps obtained by adding the individual performance measured in the single NIC experiment shown in Figure 5-5. Looking at the transmission traces we can see that the MCS being used is a mix between MCS 9 (PHY rate of 433 Mbps) and MCS 8 (PHY rate of 390 Mbps), always with a single spatial stream. In addition, in this case a retransmission rate of 31% is observed on channel 36, but the retry rate in channel 149 stays below 1%, as in the single NIC experiments. The different experienced retry rate explains the asymmetry observed in MP mode between channels 149 and 36. Further investigations is however necessary to identify why this asymmetry occurred only in MP mode, and not in AP mode, e.g. it could be due to an eventual interferer in channel 36.
- Finally, looking at the lower part of Figure 5-6 we can see that the size of the MAC aggregates in the dual NIC experiment becomes quite erratic for both the AP and MP modes. We argue that the higher number of retransmissions severely affects the AMPDU aggregation heuristics and affect MAC efficiency.

[9] This value is measured at the sniffer and not at the receiver, but we consider it indicative of the observed degradation

Figure 5-8 depicts the results of the same test performed now in crossed mode instead of direct mode (c.f. Figure 5-4). That is, each node is both TX and RX. The main findings from this experiment are:

- In AP/STA mode, the aggregate throughput decreases to 300 Mbps, as compared to the 320 Mbps available in the direct mode test. In MP mode the aggregate throughput also adds up to 300 Mbps, down from the 375 Mbps achieved in direct mode. The reason for this reduced aggregate performance in crossed mode is an increase in the frame error rate, induced by collocating a transmitter and a receiver in the same device (notice that this does not happen in direct mode, where a device either have two transmitters or two receivers).
- In addition, in crossed mode the asymmetry between the two paths further increases, with channel 149 getting significantly more bandwidth than channel 36, in both configurations but especially in MP mode. As in the direct mode experiment, this asymmetry is related to a higher number of retransmissions in channel 36. We are currently investigating the reasons for this asymmetry, which could be due to internal board effects (e.g. contention on the PCIe bus), or due to a surge of external channel interference in channel 36 when we performed the experiment. The investigation will continue in WP5.

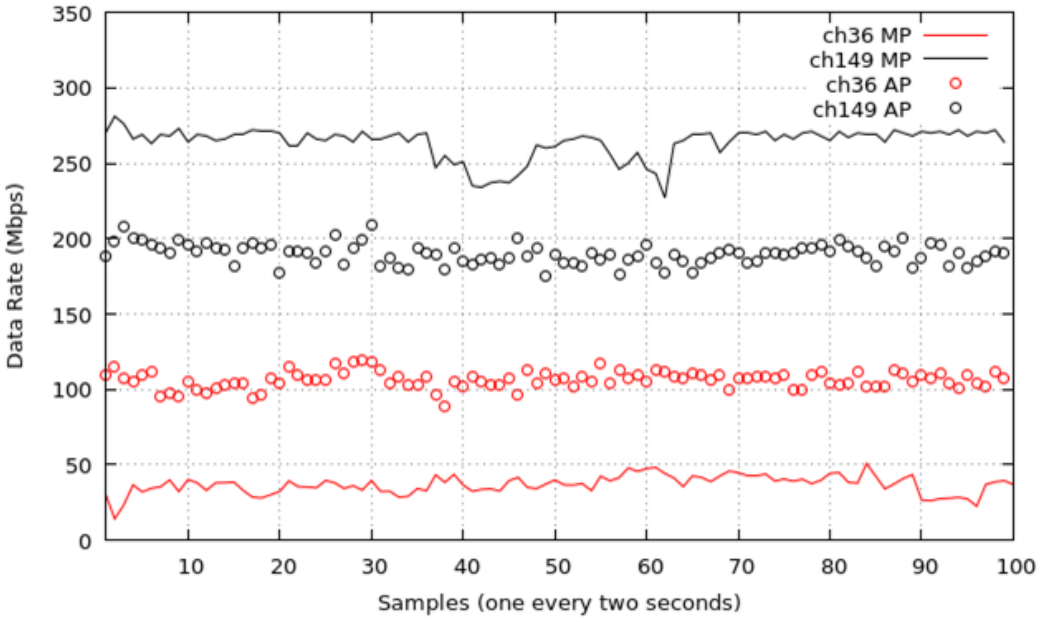


Figure 5-8. Dual NIC experiment in crossed mode.

Next, we study the effect of reducing TX power on the performance of the dual NIC scenario. In particular, we repeat the previous experiments for a 6dBm TX power, whereas all previous experiments were run at 20dBm TX power[10] (the maximum power supported by the NICs).

Figure 5-9 and Figure 5-10 report the obtained results in direct and crossed mode respectively, showing in the same plot both the 20dBm and 6dBm cases. For simplicity, the results shown correspond to the MP configuration. The main findings from the experiments are as follows:

- The obtained aggregate throughput is similar for 20dBm and 6dBm TX power, both for direct and crossed mode. When 6dBm TX power is used, we observe that the MPs mostly used MCS 8 with one spatial stream (PHY rate of 390 Mbps), which is similar than the MCS used for the 20 dBm TX power. These results, however, need to be confirmed in an outdoor setting with higher distances between transmitter and receiver, where reducing TX power may have a more dramatic effect.

[10] In some cases, the wireless NIC may be adapting power, for example to avoid PA distortion when using high order modulations. This adaptation is out of our control.

- Interestingly, a significant side effect observed is that reducing TX power seems to decrease the asymmetry observed between channels 36 and 149, both in the direct and crossed configurations. A possible reason for this is that the reduced TX power results in less power leaked from one NIC to the other, thus reducing the number of corrupted frames. We will validate these conclusions in an outdoor measurement campaign carried out in WP5.

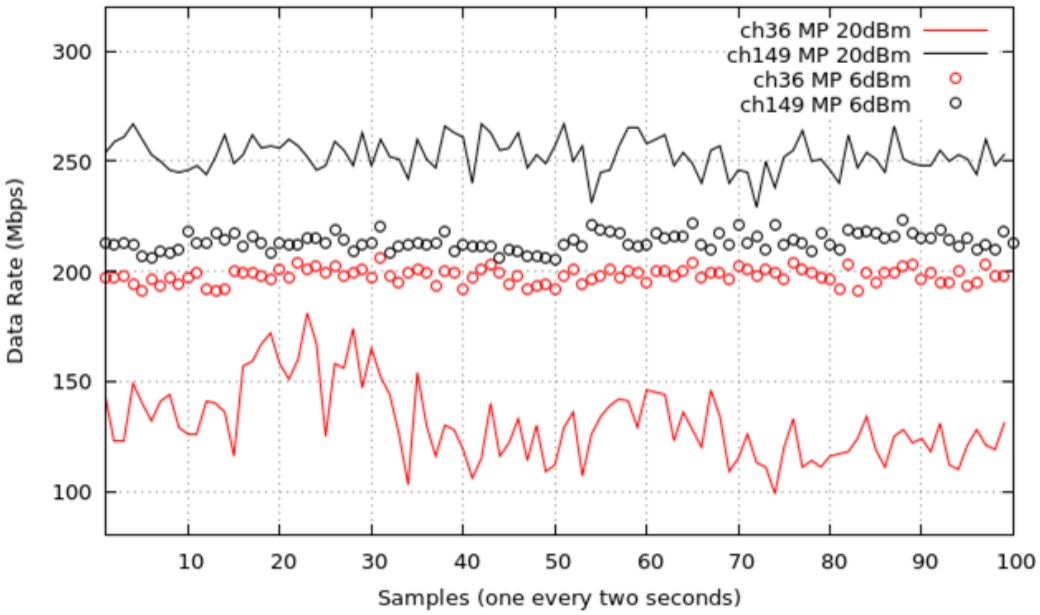


Figure 5-9. Dual NIC experiment in direct mode (20dBm and 6dBm).

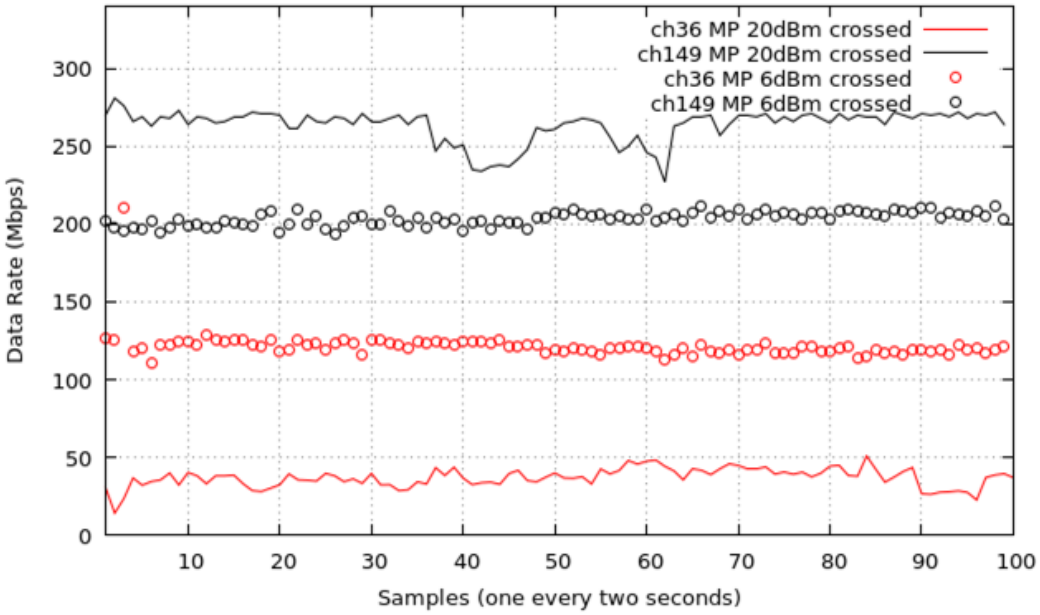


Figure 5-10. Dual NIC experiment in cross mode (20dBm and 6dBm).

5.2.3 Triple NIC experiments

Finally, we perform the triple NIC experiment described in Figure 5-4, where the two 802.11ac NICs, operating in 80 MHz mode on channels 36 and 149 of the 5 GHz band, and the 802.11n NIC, operating in 40 MHz mode in the 2.4 GHz band, are simultaneously active.

Figure 5-11 depicts the results obtained comparing the direct mode (solid lines) with the crossed mode (circles). The main findings from this experiment are:

- In direct mode, we observe an aggregate backhaul traffic (adding channels 36 and 149) around 360 Mbps, which is close to the 375 Mbps observed in the dual NIC experiment in direct mode, from which we conclude that the 5G-XHaul Sub-6 device can receive parallel transmissions in the 2.4 GHz and the 5 GHz bands without significant performance degradation. In addition, the performance of the 802.11n stream in the 2.4 GHz band is also very close to the one reported in the single NIC experiment, when the 802.11n stream transmitted in isolation. In this experiment the 5G-XHaul Sub-6 device is forwarding an overall throughput close to 450 Mbps. A fact that deserves further investigation is that the backhaul asymmetry observed in the dual NIC experiment between channels 36 and 149 (configured as MP) does not seem to be so relevant now. These phenomena will be further investigated in WP5.
- The aggregate backhaul performance in crossed mode (adding channels 36 and 149) is close to 270 Mbps, which is slightly lower than the approximately 300 Mbps obtained in the dual NIC experiments in crossed mode. The same effect is observed in the 802.11n access stream in the 2.4 GHz band that degrades from about 80 Mbps in direct mode to below 50 Mbps in crossed mode. From these results, we argue that a 5GHz transmitter collocated with a 2.4 GHz receiver degrades the performance of the 2.4 GHz receiver, and the opposite also seems to hold. Interestingly, the 5 GHz channel most affected by the 2.4 GHz transmission is channel 36 (located in the lower part of the band), while channel 149 seems mostly unaffected. This fact can be used for a proper channel allocation in the joint wireless access and backhaul deployments.

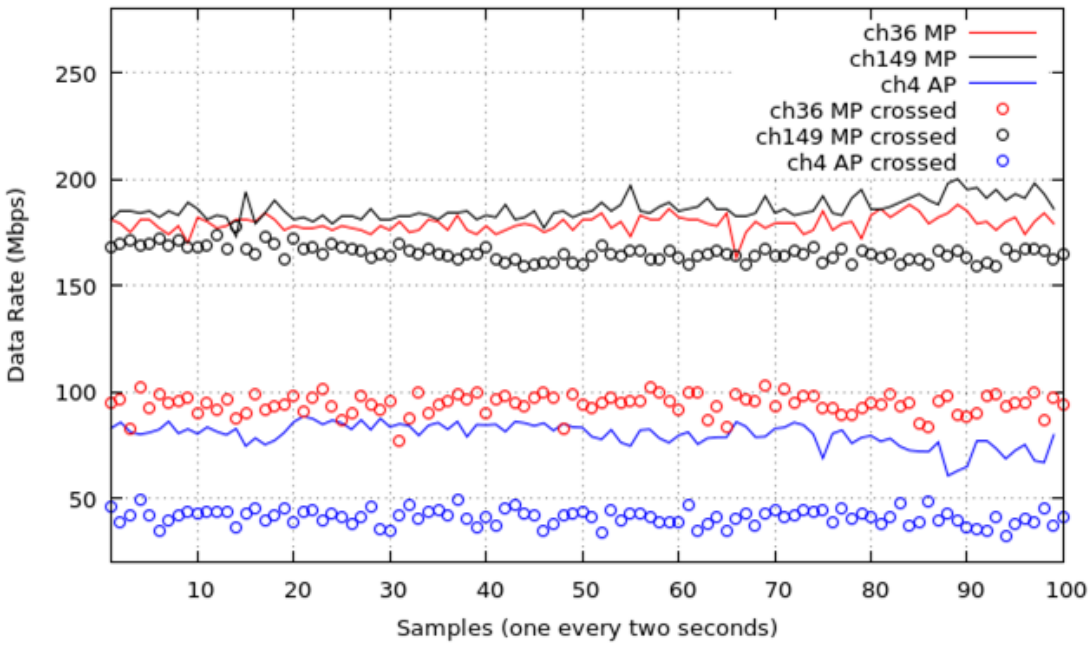


Figure 5-11. Triple NIC experiments in direct and crossed mode.

5.3 Summary and next steps

In this section, we have introduced an initial characterization of the performance to be expected from the 5G-XHaul Sub-6 devices. This performance analysis will continue within WP5, but some initial conclusions can already be derived:

- The use of spatial multiplexing is only available in AP mode, and appears to be very sensitive to external interference (observed in single NIC experiments). Hence, pending further validation in outdoor scenarios, we suspect that high order MIMO modulations defined in IEEE 802.11ac may not be available for our scenarios of interest in the short term. A software bug in the used Linux stack likely causes this problem. Thus, we expect peak link performances of around 250 Mbps (at the application layer).
- When simultaneously transmitting with two 802.11ac NICs operating at 80 MHz on opposite sides of the 5 GHz band cross-NIC interference appears, which reduces the total aggregated bandwidth to approximately 350 Mbps. This bandwidth is not always balanced between the two backhaul interfaces. Interference is worse in crossed mode (forwarding node) than in direct mode (NIC bundling).
- When adding an access interface in the 2.4GHz band, interference between access and backhaul may appear when the backhaul is in crossed mode, even using different bands (2.4 vs. 5 GHz).

In this regard, our next steps will include:

- Validate these findings in an outdoor scenario with a 50-meter distance between devices.
- Investigate the effect of larger antenna separation on cross-NIC interference.
- Investigate sources of MAC inefficiency, and study if it is possible to increase the size of MAC aggregates.
- Investigate the use of per-frame dynamic bandwidth selection in IEEE 802.11ac NICs.

6 Summary and Conclusions

This document describes the internal architecture of 5G-XHaul's Sub-6GHz transport nodes and the programmable capabilities they expose to the control plane of the network, following the SDN paradigm. The proposed architecture allows wireless TNs to be remotely controlled as wired switches in an OpenFlow-enabled network, so that the controller knows the physical topology of the wireless mesh backhaul and receives information on the status of all the nodes and links.

Thanks to the information provided by those Sub-6 GHz nodes, the control plane is aware of the status of the devices in the BH (e.g. CPU load), aware of the status of the wireless channels (e.g. interference affecting BH links), the status of all the links between neighbouring TNs (PHY bitrate, number of retransmissions, received signal strength, etc.) and the status of the flows going through those links. All that information allows the control plane to intelligently manage the (scarce) radio resources or to create optimal forwarding paths from the access to the core network dynamically, according to the changing conditions of the Sub-6 GHz wireless medium and applying a wide-range of criteria.

The implementation of the proposed architecture in different platforms allowed us to test those capabilities as a proof-of-concept. With different experiments, we showed that:

- Reporting of radio parameters is essential to properly manage a wireless backhaul infrastructure; for example, it allows interference-aware routing.
- Detection of link failures can be implemented on the BH to rapidly apply any established recovery procedure, even before the control plane becomes aware.
- Enabling TDMA scheduling on BH links allows precise QoS provision and will enable advanced services such as network slicing.

Additionally, a definite hardware platform is proposed, consisting in a SBC where we deploy the wireless SDN architecture envisioned for 5G-XHaul's Sub-6 GHz TNs. Those nodes are equipped with two IEEE 802.11ac NICs for backhaul and one IEEE 802.11n for either access or backhaul. According to our initial evaluation in an indoor testbed, the TNs built using this platform on its current form provide around 250 Mbps links in the 5GHz band and near 80 Mbps in the 2.4 GHz band. However, several issues affecting the performance of such links need further study (as part of Work Package 5). Recall that Wi-Fi-based Sub-6 GHz devices are expected to achieve the Gbps scale with PHY configurations not yet available in the market.

The progress reported in this document facilitate the integration of IEEE 802.11-based Sub-6 GHz transport nodes within the converged wireless-optical architecture envisioned by the 5G-XHaul project and constitute the required previous step towards the design and evaluation of novel intelligent and scalable control plane strategies that will be studied in Work Package 3 (WP3).

7 References

[1] 5G-XHaul, “D2.2: System Architecture Description”, July 2016.

[2] The OpenFlow Switch Specification. Available online at: https://opennetworking.org

[3] Hurtado-Borras, A., Pala-Sole, J., Camps-Mur, D., & Sallent-Ribes, S. (2015, June). SDN wireless backhauling for Small Cells. In IEEE International Conference on Communications (ICC), pp. 3897-3902, 2015.

[4] Hiertz, G.R., Denteneer, D., Max, S., Taori, R., Cardona, J., Berlemann, L. and Walke, B. IEEE 802.11 s: the WLAN mesh standard. IEEE Wireless Communications, 17(1), pp.104-111, 2010.

[5] Zehl, S. and Zubow, A. and Wolisz, A., hMAC: Enabling Hybrid TDMA/CSMA on IEEE 802.11 Hardware. Technical Report TKN-16-004, Telecommunication Networks Group, Technische Universität Berlin, 2016

[6] The OpenDayLight Platform. Available online at: https://www.opendaylight.org

[7] 5G-XHaul, “D3.1: Analysis of state of the art on scalable control plane design and techniques for user mobility awareness. Definition of 5G-XHaul control plane requirements”, July 2016.

[8] Vipin, M., and Srikanth, S. (2010, January). Analysis of open source drivers for IEEE 802.11 WLANs. In IEEE International Conference on Wireless Communication and Sensor Computing, 2010. ICWCSC 2010, (pp. 1-5).

[9] Zeng, Yunze, Parth H. Pathak, and Prasant Mohapatra. "A first look at 802.11 ac in action: energy efficiency and interference characterization." *Networking Conference, 2014 IFIP*. IEEE, 2014.

[10] J. Dixon, BT Group Strategy, “5 GHz band”, Sept. 2016. Available online: http://www.slideshare.net/TechUK/johnny-dixon-bt-and-5ghz-topic-lead-5ghz-band

[11] Nachtigall, J., Zubow, A. and Redlich, J-P. "The impact of adjacent channel interference in multi-radio systems using IEEE 802.11." IEEE International Wireless Communications and Mobile Computing Conference, 2008.

[12] Xia, D., Hart, J. and Fu, Q. "Evaluation of the Minstrel rate adaptation algorithm in IEEE 802.11 g WLANs." *2013 IEEE International Conference on Communications (ICC)*. IEEE, 2013.

8 Acronyms

Acronym	Description
3GPP	Third Generation Partnership Project
5G	Fifth Generation Networks
ACK	Acknowledgment frame/packet
AN	Access Network node
AP	Access Point
BH	Backhaul
BI	Beacon Interval
CSMA	Carrier-Sense Multiple Access
DCF	Distributed Coordination Function
DFS	Dynamic Frequency Selection
EDCA	IEEE 802.11e's Enhanced distributed channel access
FH	Fronthaul
ISM	Industrial Scientific Medical (license-free frequency bands)
LLID	Local Link Identifier (in PLM)
LoS	Line-of-Sight
MAC	Medium Access Control
MCS	Modulation and Coding Scheme
MIMO	Multiple-Input Multiple-Output
MLME	Media Access Control (MAC) Sublayer Management Entity
mmWave	Millimeter Wave
NIC	Network Interface Card
NLoS	Non-Line-of-Sight
P2P	Point-to-Point
P2MP	Point-to-Multipoint
PHY	Physical layer
PLID	Peer Link Identifier (in PLM)
PLM	Peer Link Management
PTP	Precision Time Protocol
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service
RAN	Radio Access Network
RAT	Radio Access Technology
RF	Radio Frequency
RSSI	Received Signal Strength Indicator

RX	Receiver/Reception/Received depending on context
SBC	Single Board Computer
SDN	Software Defined Networking
SINR	Signal to Interference plus Noise Ratio
TC	Transport Class
TDMA	Time Division Multiple Access
TN	Transport Node
TU	Time Unit (1024 μs)
TX	Transmitter/Transmission/Transmitted depending on context
VNF	Virtual Network Functions
VNO	Virtual Network Operator
VNP	Virtual Network Provider
vCN	virtualised Core Network
vRAN	Virtualised RAN