Cellular Networks

A Systems Approach

Larry Peterson and Oguz Sunay

Version 0.1

Table of Contents

README

This repository contains source for *Cellular Networks: A Systems Approach*, available under terms of the Creative Commons (CC BY 4.0) license. The community is invited to contribute corrections, improvements, updates, and new material under the same terms.

If you make use of this work, the attribution should include the following information:

> *Title: Cellular Networks: A Systems Approach*
> *Authors: Larry Peterson and Oguz Sunay*
> *Source: https://github.com/SystemsApproach/5G*
> *License: CC BY 4.0*

Build the Book

To build a web-viewable version, you first need to install a couple packages:

- Gitbook Toolchain
- Node.js Package Manager

Then do the following to download the source:

```
mkdir ~/5G
cd ~/5G
git clone https://github.com/SystemsApproach/5G.git
cd 5G
```

To build a web version of the book, simply type:

```
make
```

If all goes well, you will be able to view the book in your browser at `localhost:4000`. (If all doesn't go well, you might try typing `make` a second time.)

Preface

The transition to 5G is happening, and unless you've been living under a rock, you've surely heard the hype. But if you are like 99% of the CS-trained, systems-oriented, cloud-savvy, IETF-participating people in the world, the cellular network is largely a mystery. You know it's an important technology used in the last mile to connect people to the Internet, but you've otherwise abstracted it out of your scope-of-concerns.

The important thing to understand about 5G is that it implies much more than a generational upgrade in bandwidth. It involves transformative changes that blur the line between the access network and the cloud. And it will encompass enough value that it has the potential to turn the "Access-as-part-of-Internet" perspective on its head. We will just as likely be talking about "Internet-as-backend-to-Access" ten years from now.

This book is written for someone that has a working understanding of the Internet and Cloud, but has had limited success penetrating the myriad of acronyms that dominate cellular networking. In fairness, the Internet has its share acronyms, but it also comes with a sufficient set of abstractions to help manage the complexity. It's hard to say the same for the cellular network, where pulling on one thread seemingly unravels the entire space. It has also been the case that the cellular network had been largely hidden inside proprietary devices, which has made it impossible to figure it out for yourself.

This book is the result of a cellular networking expert teaching a systems person about 5G as we've collaborated on an open source 5G implementation. The material has been used to train other software developers, and we are hopeful it will be useful to anyone that wants a deeper understanding of 5G and the opportunity for innovation it provides. Readers that want hands-on experience can also access the open source software introduced in the book.

This book will likely be a work-in-progress for the foreseeable future. It's not intended to be encyclopedic—favoring perspective and end-to-end completeness over every last bit of detail—but we do plan to flesh out the content over time. Your suggestions (and contributions) to this end are welcome.

Larry Peterson and Oguz Sunay
Open Networking Foundation
August 2019

Chapter 1: Introduction

Cellular networks, which have a 40-year history that parallels the Internet's, have undergone significant change. The first two generations supported voice and then text, with 3G defining the transition to broadband access, supporting data rates measured in hundreds of kilobits-per-second. Today, the industry is at 4G (supporting data rates typically measured in the few megabits-per-second) and starting the transition to 5G, with the promise of a tenfold increase in data rates.

But 5G is about much more than increased bandwidth. In the same way 3G defined the transition from voice to broadband, 5G's promise is mostly about the transition from a single access service (broadband connectivity) to a richer collection of edge services and devices, including support for immersive user interfaces (e.g., AR/VR), mission-critical applications (e.g., public safety, autonomous vehicles), and the Internet-of-Things (IoT). Because these use cases will include everything from home appliances to industrial robots to self-driving cars, 5G won't just support humans accessing the Internet from their smartphones, but also swarms of autonomous devices working together on their behalf. All of this requires a fundamentally different architecture.

This book describes the 5G cellular network, which because it is on an evolutionary path and not a point solution, includes standardized specifications, a range of implementation choices, and a long list of aspirational goals. Because this leaves so much room for interpretation, our approach to describing 5G is grounded in two mutually-supportive principles. The first is to apply a *systems lens*, which is to say, we explain the sequence of design decisions that lead to a solution rather than fall back on enumerating the overwhelming number of acronyms as a *fait accompli*. The second is to aggressively disaggregate the system. Building a disaggregated, virtualized, and software-defined 5G access network is the direction the industry is already headed (for good technical and business reasons), but breaking the 5G network down into its elemental components is also the best way to explain how 5G works. It also helps to illustrate how 5G might evolve in the future to provide even more value.

What this all means is that there is no single, comprehensive definition of 5G, any more than there is for the Internet. It is a complex and evolving system, constrained by a set of standards that purposely give all the stakeholders many degrees of freedom. In the chapters that follow, it should be clear from the context whether we are talking about *standards* (what everyone must do), *trends* (where the industry seems to be headed), or *implementation choices* (examples to make the discussion more concrete). By adopting a systems perspective throughout, our intent is to describe 5G in a way that helps the reader navigate this rich and rapidly evolving system.

Standardization Landscape

As of 3G, the generational designation corresponds to a standard defined by the 3GPP (3rd Generation Partnership Project). Even though its name has "3G" in it, the 3GPP continues to define the standard for 4G and 5G, each of which corresponds to a sequence of releases of the standard. Release 15 is considered the demarcation point between 4G and 5G, with Release 16 expected by the end of 2019. Complicating the terminology, 4G was on a multi-release evolutionary path referred to as *Long Term Evolution (LTE)*. 5G is on a similar evolutionary path, with several expected releases over its lifetime.

While 5G is an ambitious advance beyond 4G, it is also the case that understanding 4G is the first step to understanding 5G, as several aspects of the latter can be explained as bringing a new degree-of-freedom to the former. In the chapters that follow, we often introduce some architectural feature of 4G as a way of laying the foundation for the corresponding 5G component.

Like Wi-Fi, cellular networks transmit data at certain bandwidths in the radio spectrum. Unlike Wi-Fi, which permits anyone to use a channel at either 2.4 or 5 GHz (these are unlicensed bands), governments have auctioned off and licensed exclusive use of various frequency bands to service providers, who in turn sell mobile access service to their

subscribers.

There is also a shared-license band at 3.5-GHz, called *Citizens Broadband Radio Service (CBRS)*, set aside in North America for cellular use. Similar spectrum being set aside in other countries. The CBRS band allows three tiers of users to share the spectrum: first right of use goes to the original owners of this spectrum, naval radars and satellite ground stations; followed by priority users who receive this right over 10MHz bands for three years via regional auctions; and finally the rest of the population, who can access and utilize a portion of this band as long as they first check with a central database of registered users. CBRS, along with standardization efforts to extend cellular networks to operate in the unlicensed bands, open the door for private cellular networks similar to Wi-Fi.

The specific frequency bands that are licensed for cellular networks vary around the world, and are complicated by the fact that network operators often simultaneously support both old/legacy technologies and new/next-generation technologies, each of which occupies a different frequency band. The high-level summary is that traditional cellular technologies range from 700-MHz to 2400-MHz, with new mid-spectrum allocations now happening at 6-GHz, and millimeter-wave (mmWave) allocations opening above 24-GHz.

While the specific frequency band is not directly relevant to understanding 5G from an architectural perspective, it does impact the physical-layer components, which in turn has indirect ramifications on the overall 5G system. We will identify and explain these ramifications in later chapters.

Access Networks

The cellular network is part of the access network that implements the Internet's so-called *last mile*. Other access technologies include *Passive Optical Networks (PON)*, colloquially known as Fiber-to-the-Home. These access networks are provided by both big and small network operators. Global network operators like AT&T run access networks at thousands of points-of-presence across a country like the US, along with a national backbone that interconnects those sites. Small regional and municipal network operators might run an access network with one or two points-of-presence, and then connect to the rest of the Internet through some large operator's backbone.

In either case, access networks are physically anchored at thousands of points-of-presence within close proximity to end users, each of which serves anywhere from 1,000 to 100,000 subscribers, depending on population density. In practice, the physical deployment of these ”edge” locations vary from operator to operator, but one possible scenario is to anchor both the cellular and wireline access networks in Telco *Central Offices*.

Historically, the Central Office—officially known as the *PSTN (Packet-Switched Telephone Network) Central Office*—anchored wired access (both telephony and broadband), while the cellular network evolved independently by deploying a parallel set of *Mobile Telephone Switching Offices (MTSO)*. Each MTSO serves as a *mobile aggregation* point for the set of cell towers in a given geographic area. For our purposes, the important idea is that such aggregation points exist, and it is reasonable to think of them as defining the edge of the operator-managed access network. For simplicity, we sometimes use the term “Central Office” as a synonym for both types of edge sites.

Note that in addition to these edge locations, there are also centralized components of the cellular network that track subscribers as they move from one location to another, but each individual access network site is limited in size due to the technology it employs. For example, PON and RAN both have a maximum reach measured in a few kilometers.

Edge Cloud

Because of their wide distribution and close proximity to end users, Central Offices are also an ideal place to host the edge cloud. But this begs the question: What exactly is the edge cloud?

In a nutshell, the cloud began as a collection of warehouse-sized datacenters, each of which provided a cost-effective way to power, cool, and operate a scalable number of servers. Over time, this shared infrastructure lowered the barrier to deploying scalable Internet services, but today, there is increasing pressure to offer low-latency/high-bandwidth cloud applications that cannot be effectively implemented in centralized datacenters. Augmented Reality (AR), Virtual Reality (VR), Internet-of-Things (IoT), Autonomous Vehicles are all examples of this kind of application. This has resulted in a trend to move some functionality out of the datacenter and towards the edge of the network, closer to end users.

Where this edge is *physically* located depends on who you ask. If you ask a network operator that already owns and operates thousands of Central Offices, then their Central Offices are an obvious answer. Others might claim the edge is located at the 14,000 Starbucks across the US, and still others might point to the tens-of-thousands of cell towers spread across the globe.

Our approach is to be location agnostic, but it is worth pointing out that the cloud's migration to the edge coincides with a second trend, which is that network operators are re-architecting the access network to use the same commodity hardware and best practices in building scalable software as the cloud providers. Such a design, which is sometimes referred to as *CORD (Central Office Re-architected as a Datacenter)*, supports both the access network and edge services co-located on a shared cloud platform. This platform is then replicated across hundreds or thousands of sites (including, but not limited to, Central Offices). So while we shouldn't limit ourselves to the Central Office as the only answer to the question of where the edge cloud is located, it is becoming a viable option.

When we get into the details of how 5G can be implemented in practice, we use CORD as our exemplar. For now, the important thing to understand is that 5G is being implemented as software running on commodity hardware, rather than embedded in the special-purpose proprietary hardware used in past generations. This has a significant impact on how we think about 5G (and how we describes 5G), which will increasingly become yet another software-based component in the cloud, as opposed to an isolated and specialized technology attached to the periphery of the cloud.

> Our use of CORD as an exemplar is not to imply that the edge cloud is limited to Central Offices. CORD is a good exemplar because it is designed to host both edge services and access technologies like 5G on a common platform, where the Telco Central Office is one possible location to deploy such a platform.

An important takeaway from this discussion is that to understand how 5G is being implemented, it is helpful to have a working understanding of how clouds are built. This includes the use of *commodity hardware* (both servers and white-box switches), horizontally scalable *microservices* (also referred to as *cloud native*), and *Software-Defined Networks (SDN)*. is also helpful to have an appreciation for how cloud software is developed, tested, deployed and operated, including practices like *DevOps* and *Continuous Integration / Continuous Deployment (CI/CD)*.

One final note about terminology. Anyone that has been paying attention to the discussion surrounding 5G will have undoubtedly heard about *Network Function Virtualization (NFV)*, which involves moving functionality that was once embedded in hardware appliances into VMs running on a commodity server. In our experience, NFV is a stepping stone towards the fully disaggregated solution we describe, and so we do not dwell on it. You can think of NFV as an alternative to the cloud native exemplar presented here.

Chapter 2: Wireless Transmission

For anyone familiar with wireless access technologies like Wi-Fi, the cellular network is most unique due to its approach to sharing the available radio spectrum among its many users, all the while allowing those users to remain connected while moving. This has resulted in a highly dynamic and adaptive approach, in which coding, modulation and scheduling play a central role.

As we will see later in this chapter, cellular networks use a reservation-based strategy, whereas Wi-Fi is contention-based. This difference is rooted in each system's fundamental assumption about utilization: Wi-Fi assumes a lightly loaded network (and hence optimistically transmits when the wireless link is idle and backs off if contention is detected), while 4G and 5G cellular networks assume (and strive for) high utilization (and hence explicitly assign different users to different "shares" of the available radio spectrum).

We start by giving short primer on radio transmission as a way of laying a foundation for understanding the rest of the 5G architecture. The following is not a substitute for a theoretical treatment of the topic, but is instead intended as a way of grounding the systems-oriented description of 5G that follows in the reality of wireless communication.

Coding and Modulation

At its core, coding inserts extra bits into the data to account for environmental factors that interfere with signal propagation, and modulation translates the encoded data into electromagnetic signals by varying the signal's amplitude and phase. This relationship is depicted in Figure 2.1, where coding typically implies some form of *Forward Error Correction* (e.g., turbo codes, polar codes) and modulation is done relative to an assigned *carrier signal*.

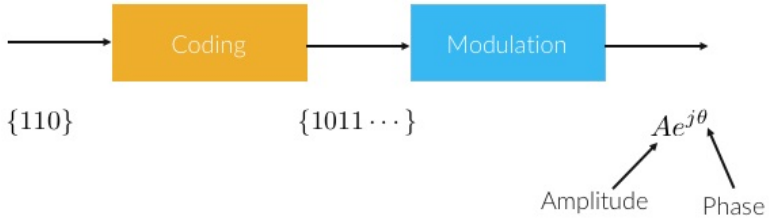

Figure 2.1: The role of coding and modulation in wireless communication.

When dealing with radio signals propagating through the air, opportunities for interference abound. As illustrated in Figure 2.2, the signal bounces off various stationary and moving objects, following multiple paths from the transmitter to the receiver, who may also be moving.

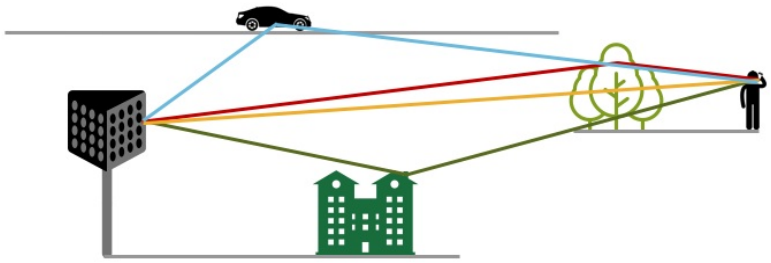

Figure 2.2: Signals propagate along multiple paths from transmitter to receiver.

As a consequence of these multiple paths, the original signal arrives at the receiver spread over time, as illustrated in Figure 2.3. Empirical evidence shows that the Multipath Spread—the time between the first and last signals of one transmission arriving at the receiver—is 1 to 10μs in urban environments and 10 to 30μs in suburban environments.

Theoretical bounds for the time duration for which the channel may be assumed to be time invariant, known as the Coherence Time and denoted T_c , is given by

$$T_c = c/v \times 1/f$$

where c is the velocity of the signal, v is the velocity of the receiver (e.g., moving car or train), and f is the frequency of the carrier signal that is being modulated. This says the coherence time is inversely proportional to the frequency of the signal and the speed of movement, which makes intuitive sense: The higher the frequency (narrower the wave) the shorter the coherence time, and likewise, the faster the receiver is moving the longer the coherence time. Based on the target parameters to this model (selected according to the target physical environment), it is possible to calculate T_c which in turn bounds the rate at which symbols can be transmitted without undue risk of interference.

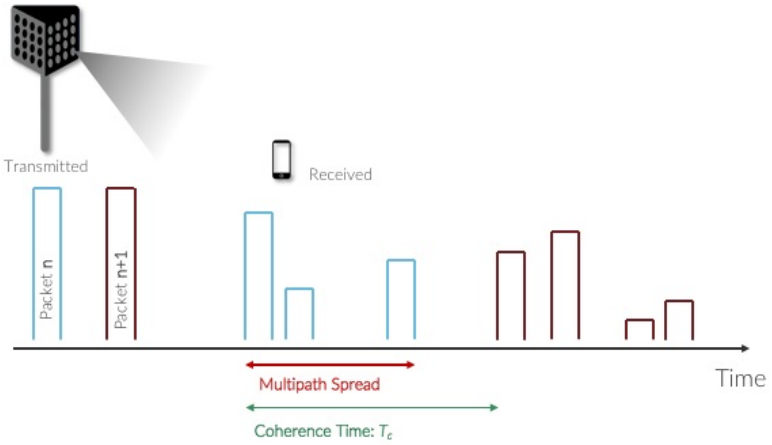

Figure 2.3: Received data spread over time due to multipath variation.

To complicate matters further, Figures 2.2 and 2.3 imply the transmission originates from a single antenna, but cell towers are equipped with an array of antennas, each transmitting in a different (but overlapping) direction. This technology, called *Multiple-Input-Multiple-Output (MIMO)*, opens the door to purposely transmitting data from multiple antennas in an effort to reach the receiver, adding even more paths to the environment-imposed multipath propagation.

One of the most important consequences of these factors is that the transmitter must receive feedback from every receiver to judge how to best utilize the wireless medium on their behalf. 3GPP specifies a *Channel Quality Indicator (CQI)* for this purpose, where in practice the receiver sends a CQI status report to the base station periodically (e.g., every millisecond). These CQI messages report the observed signal-to-noise ratio, which impacts the receiver's ability to recover the data bits. The base station then uses this information to adapt how it allocates the available radio spectrum to the subscribers it is serving. This allocation decision is the job of the scheduler.

How the scheduler does its job is one of the most important properties of each generation of the cellular network, which in turn depends on the multiplexing mechanism. For example, 2G used *Time Division Multiple Access (TDMA)* and 3G used *Code Division Multiple Access (CDMA)*. It is also a major differentiator for 4G and 5G, completing the transition from the cellular network being fundamentally circuit-switched to fundamentally packet-switched. The following two sections describe each, in turn.

Scheduling: 4G

The state-of-the-art in multiplexing 4G cellular networks is called *Orthogonal Frequency-Division Multiple Access (OFDMA)*. The idea is to multiplex data over a set of 12 orthogonal subcarrier frequencies, each of which is modulated independently. The "Multiple Access" in OFDMA implies that data can simultaneously be sent on behalf of multiple

users, each on a different subcarrier frequency and for a different duration of time. The subbands are narrow (e.g., 15kHz), but the coding of user data into OFDMA symbols is designed to minimize the risk of data loss due to interference between adjacent bands.

The use of OFDMA naturally leads to conceptualizing the radio spectrum as a two-dimensional resource, as shown in Figure 2.4. The minimal schedulable unit, called a *Resource Element (RE)*, corresponds to a 15kHz-wide band around one subcarrier frequency and the time it takes to transmit one OFDMA symbol. The number of bits that can be encoded in each symbol depends on the modulation rate, so for example using *Quadrature Amplitude Modulation (QAM)*, 16-QAM yields 4 bits per symbol and 64-QAM yields 16 bits per symbol

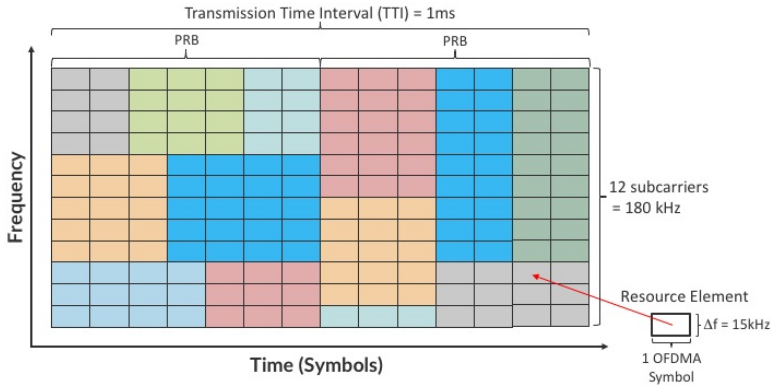

Figure 2.4: Spectrum abstractly represented by a 2-D grid of schedulable Resource Elements.

A scheduler allocates some number of REs to each user that has data to transmit during each 1ms *Transmission Time Interval (TTI*, where users are depicted by different colored blocks in Figure 4. The only constraint on the scheduler is that it must make its allocation decisions on blocks of 7x12=84 resource elements, called a *Physical Resource Block (PRB)*. Figure 2.4 shows two back-to-back PRBs. Of course time continues to flow along one axis, and depending on the size of the available frequency band (e.g., it might be 100MHz wide), there may be many more subcarrier slots (and hence PRBs) available along the other axis, so the scheduler is essentially preparing and transmitting a sequence of PRBs.

Note that OFDMA is not a coding/modulation algorithm, but instead provides a framework for selecting a specific coding and modulator for each subcarrier frequency. QAM is one common example modulator. It is the scheduler's responsibility to select the modulation to use for each PRB, based on the CQI feedback it has received. The scheduler also selects the coding on a per-PRB basis, for example, by how it sets the parameters to the turbo code algorithm.

The 1ms TTI corresponds to the time frame in which the scheduler receives feedback from users about the quality of the signal they are experiencing. This is the CQI mentioned earlier, where once every millisecond, each user sends a set of metrics, which the scheduler uses to make its decision as to how to allocate PRBs during the subsequent TTI.

Another input to the scheduling decision is the *QoS Class Identifier (QCI)*, which indicates the quality-of-service each class of traffic is to receive. In 4G, the QCI value assigned to each class (there are nine such classes, in total) indicates whether the traffic has a *Guaranteed Bit Rate (GBR)* or not *(non-GBR)*, plus the class's relative priority within those two categories.

Finally, keep in mind that Figure 2.4 focuses on scheduling transmissions from a single antenna, but the MIMO technology described above means the scheduler also has to determine which antenna (or more generally, what subset of antennas) will most effectively reach each receiver. But again, in the abstract, the scheduler is charged with allocating a sequence of Resource Elements.

This all begs the question: How does the scheduler decide which set of users to service during a given time interval, how many resource elements to allocate to each such user, how to select the coding and modulation levels, and which antenna to transmit their data on? This is an optimization problem that, fortunately, we are not trying to solve

here. Our goal is to describe an architecture that allows someone else to design and plug in an effective scheduler. Keeping the cellular architecture open to innovations like this is one of our goals, and as we will see in the next section, becomes even more important in 5G where the scheduler operates with even more degrees of freedom.

Scheduling: 5G

The transition from 4G to 5G introduces additional degrees-of-freedom in how the radio spectrum is scheduled, making it possible to adapt the cellular network to a more diverse set of devices and applications domains.

Fundamentally, 5G defines a family of waveforms—unlike LTE, which specified only one waveform—each optimized for a different band in the radio spectrum. (A waveform is the frequency, amplitude, and phase-shift independent property (shape) of a signal. A sine wave is an example waveform.) The bands with carrier frequencies below 1GHz are designed to deliver mobile broadband and massive IoT services with a primary focus on range. Carrier frequencies between 1GHz-6GHz are designed to offer wider bandwidths, focusing on mobile broadband and mission-critical applications. Carrier frequencies above 24GHz (mmWaves) are designed to provide super wide bandwidths over short, line-of-sight coverage.

These different waveforms affect the scheduling and subcarrier intervals (i.e., the “size” of the resource elements described in the previous section).

- For sub-1GHz bands, 5G allows maximum 50MHz bandwidths. In this case, there are two waveforms: one with subcarrier spacing of 15kHz and another of 30kHz. (We used 15kHz in the example shown in Figure 2.4.) The corresponding scheduling intervals are 0.5ms and 0.25ms, respectively. (We used 0.5ms in the example shown in Figure 2.4.)
- For 1GHz-6GHz bands, maximum bandwidths go up to 100MHz. Correspondingly, there are three waveforms with subcarrier spacings of 15kHz, 30kHz and 60kHz, corresponding to scheduling intervals of 0.5ms, 0.25ms and 0.125ms, respectively.
- For millimeter bands, bandwidths may go up to 400MHz. There are two waveforms, with subcarrier spacings of 60kHz and 120kHz. Both have scheduling intervals of 0.125ms.

This range of options is important because it adds another degree of freedom to the scheduler. In addition to allocating radio resources to users, it has the ability to dynamically adjust the size of the resource by changing the wave form being used. With this additional freedom, fixed-sized REs are no longer the primary unit of resource allocation. We instead use more abstract terminology, and talk about allocating *Resource Blocks* to subscribers, where the 5G scheduler determines both the size and number of Resource Blocks allocated during each time interval.

Figure 2.5 depicts the role of the scheduler from this more abstract perspective, where just as with 4G, CQI feedback from the receivers and the QCI quality-of-service class selected by the subscriber are the two key pieces of input to the scheduler. Note that the set of QCI values changes between 4G and 5G, reflecting the increasing differentiation being supported. For 5G, each class includes the following attributes:

- Resource Type: Guaranteed Bit Rate (GBR), Delay-Critical GBR, Non-GBR
- Priority Level
- Packet Delay Budget
- Packet Error Rate
- Averaging Window
- Maximum Data Burst

Note that while the preceding discussion could be interpreted to imply a one-to-one relationship between subscribers and a QCI, it is more accurate to say that each QCI is associated with a class of traffic (often corresponding to some type of application), where a given subscriber might be sending and receiving traffic that belongs to multiple classes at any given time. We explore this idea in much more depth in a later section.

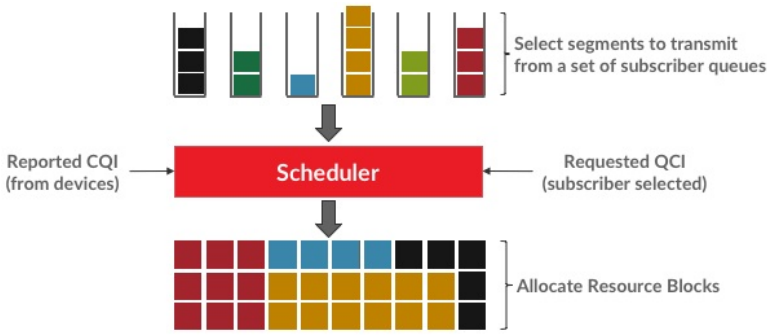

Figure 2.5: Scheduler allocates Resource Elements to user data streams based on CQI feedback from receivers and the QCI parameters associated with each class of service.

Chapter 3: Basic Architecture

This section identifies the main architectural components of cellular access networks. It focuses on the components that are common to both 4G and 5G, and as such, establishes a foundation for understanding the advanced features of 5G presented in the sections that follow.

This overview is partly an exercise in introducing 3GPP terminology. For someone that is familiar with the Internet, this terminology can seem arbitrary (e.g., “eNB” is a “base station”), but it is important to keep in mind that this terminology came out of the 3GPP standardization process, which has historically been concerned about telephony and almost completely disconnected from the IETF and other Internet-related efforts. To further confuse matters, the 3GPP terminology often changes with each generation (e.g., a base station is called eNB in 4G and gNB in 5G). We address situations like this by using generic terminology (e.g., base station), and referencing the 3GPP-specific counterpart only when the distinction is helpful.

> This example is only the tip of the terminology iceberg. For a slightly broader perspective on the complexity of terminology in 5G, see Marcin Dryjanski’s blog post.

Main Components

The cellular network provides wireless connectivity to devices that are on the move. These devices, which are known as *User Equipment (UE)*, have until recently corresponded to smartphones and tablets, but will increasingly include cars, drones, industrial and agricultural machines, robots, home appliances, medical devices, and so on.

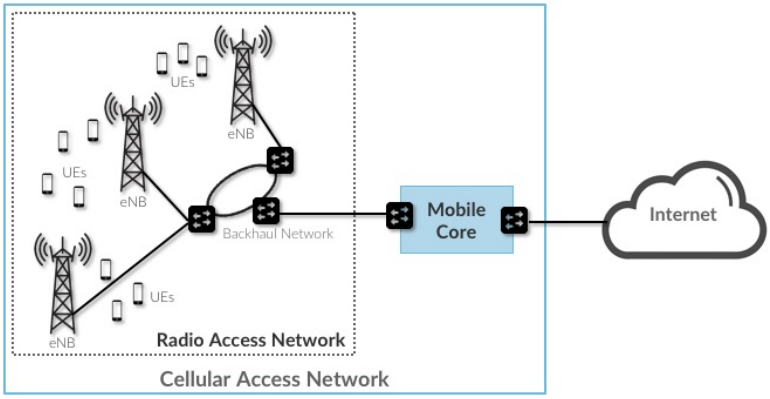

Figure 3.1: Cellular networks consists of a Radio Access Network (RAN) and a Mobile Core.

As shown in Figure 3.1, the cellular network consists of two main subsystems: the *Radio Access Network (RAN)* and the *Mobile Core*. The RAN manages the radio spectrum, making sure it is both used efficiently and meets the quality-of-service requirements of every user. The main component in the RAN is the crypticly named *eNodeB* (or *eNB*), which is short for the equally cryptic *evolved Node B*. The Mobile Core is a bundle of functionality (as opposed to a device) that serves several purposes:

- Provides Internet (IP) connectivity for both data and voice services.
- Ensures this connectivity fulfills the promised QoS requirements.
- Tracks user mobility to ensure uninterrupted service.
- Tracks subscriber usage for billing and charging.

> Mobile Core is another example of a generic term. In 4G this is called the Evolved Packet Core (EPC) and in 5G it is called the Next Generation Core (NG-Core).

Even though the word “Core” is in its name, from an Internet perspective, the Mobile Core is still part of the access network, effectively providing a bridge between the RAN and the IP-world.

Taking a closer look at Figure 3.1, we see that a *Backhaul Network* interconnects the eNBs that implement the RAN with the Mobile Core. This network is typically wired, may or may not have the ring topology shown in the Figure, and is often constructed from commodity components found elsewhere in the Internet. For example, the Passive Optical Network (PON) that implements Fiber-to-the-Home is a prime candidate for implementing the RAN backhaul. The backhaul network is obviously a necessary part of the RAN, but it is an implementation choice and not prescribed by the 3GPP standard.

Although 3GPP specifies all the elements that implement the RAN and Mobile Core in an open standard—including sub-layers we have not yet introduced—network operators have historically bought proprietary implementations of each subsystem from a single vendor. This lack of an open source implementation contributes to the perceived “opaqueness” of the cellular network in general, and the RAN in particular. And while it is true that an eNodeB implementation does contain sophisticated algorithms for scheduling transmission on the radio spectrum—algorithms that are considered valuable Intellectual Property of the equipment vendors—there is significant opportunity to open and disaggregate both the RAN and the Mobile Core. The following two sections describe each, in turn.

Before getting to those details, Figure 3.2 redraws the components from Figure 3.1 to highlight two important distinctions. The first is that the eNB (which we will refer to as the Base Station from here on) has an analog component (depicted by an antenna) and a digital component (depicted by a processor). The second is that the Mobile Core is partitioned into a *Control Plane* and *User Plane*, which is similar to the control/data plane split that someone familiar with the Internet would recognize. (3GPP also recently introduced a corresponding acronym —*CUPS, Control and User Plane Separation*—to denote this idea). The importance of these two distinctions will become clear in the following discussion.

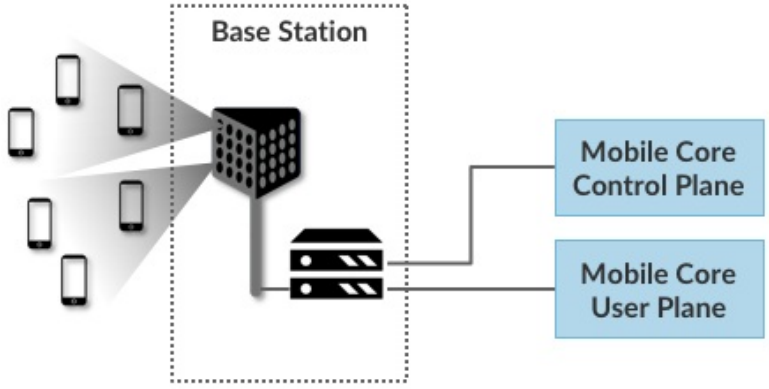

FIgure 3.2: Mobile Core divided into a Control Plan and a User Plane, an architectural feature known as CUPS: Control and User Plane Separation

Radio Access Network

We now describe the RAN by sketching the role each base station plays. Keep in mind this is kind of like describing the Internet by explaining how a router works—a not unreasonable place to start, but it doesn’t fully do justice to the end-to-end story.

First, each base station establishes the wireless channel for a subscriber’s UE upon power-up or upon handover when the UE is active. This channel is released when the UE remains idle for a predetermined period of time. Using 3GPP terminology, this wireless channel is said to provide a bearer service.

> The term “bearer” has historically been used in telecommunications (including early wireline technologies like ISDN) to denote “data,” as opposed to a channel that carries “signalling” information.

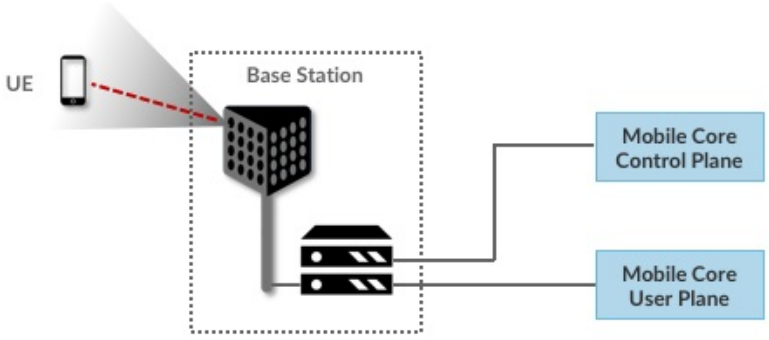

Figure 3.3: Base Station detects (and connects to) active UEs.

Second, each base station establishes ”3GPP Control Plane” connectivity between the UE and the corresponding Mobile Core Control Plane component, and forwards signaling traffic between the two. This signaling traffic enables UE authentication, registration, mobility tracking.

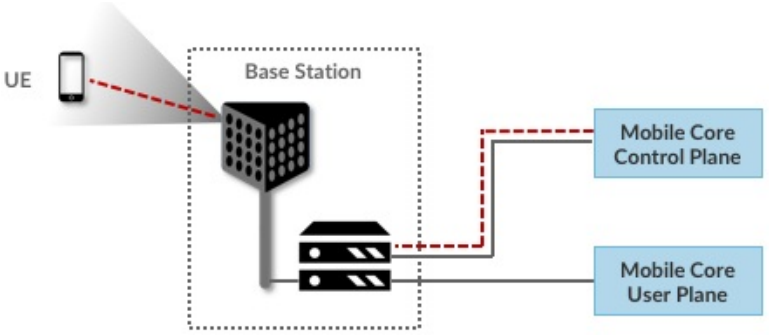

FIgure 3.4: Base Station establishes control plane connectivity between each UE and the Mobile Core.

Third, for each active UE, the base station establishes one or more tunnels between the corresponding Mobile Core User Plane component.

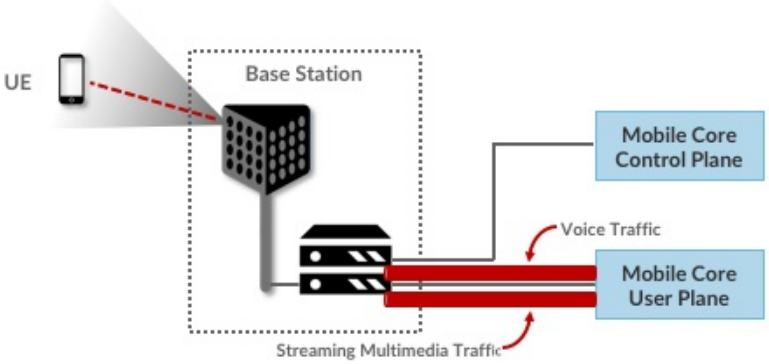

Figure 3.5: Base station establishes one or more tunnels between each UE and the Mobile Core's User Plane.

Fourth, the base station forwards both control and user plane packets between the Mobile Core and the UE. These packets are tunnelled over SCTP/IP and GTP/UDP/IP, respectively. SCTP (Stream Control Transport Protocol) is 3GPP-defined alternative to TCP, tailored to carry signalling (control) information for telephony services. GTP (a nested acronym corresponding to (General Packet Radio Service) Tunneling Protocol) is a 3GPP-specific tunneling protocol designed to run over UDP.

As an aside, it is noteworthy that connectivity between the RAN and the Mobile Core is IP-based. This was introduced as one of the main changes between 3G and 4G. Prior to 4G, the internals of the cellular network were circuit-based, which is not surprising given its origins as a voice network.

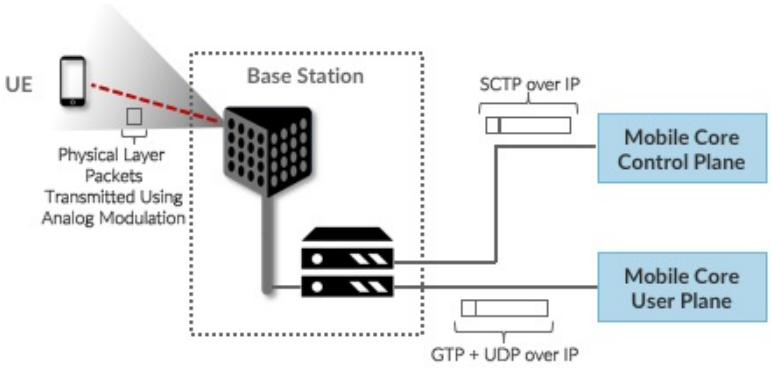

FIgure 3.6: Base Station to Mobile Core (and Base Station to Base Station) control plane tunneled over SCTP/IP and user plane tunneled over GTP/UDP/IP.

Fifth, the base station coordinates UE handovers between neighboring base stations, using direct station-to-station links. Exactly like the station-to-core connectivity shown in the previous figure, these links are used to transfer both control plane (SCTP over IP) and user plane (GTP over UDP/IP) packets.

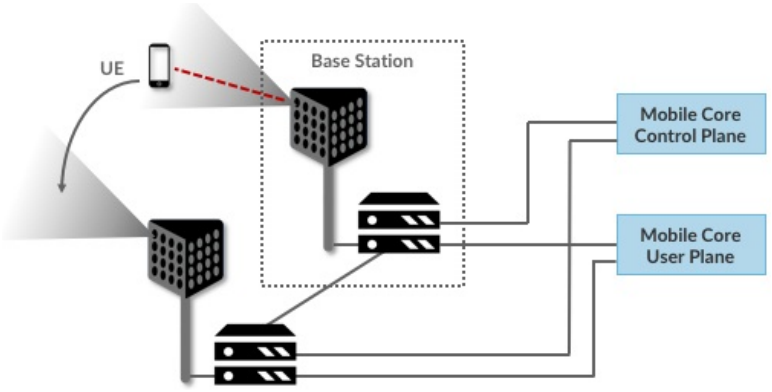

FIgure 3.7: Base Stations cooperate to implement UE hand over.

Sixth, the base station coordinates wireless multi-point transmission to a UE from multiple base stations, which may or may not be part of a UE handover from one base station to another.

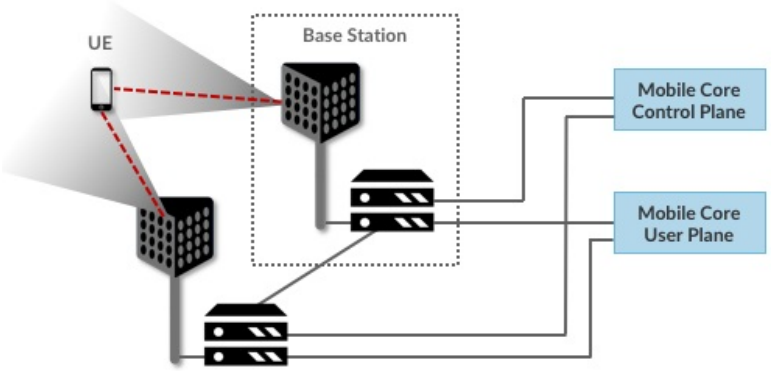

Figure 3.8: Base Stations cooperate to implement multipath transmission (link aggregation) to UEs.

For our purposes, the main takeaway is that the base station can be viewed as a specialized forwarder. In the Internet-to-UE direction, it fragments outgoing IP packets into physical layer segments and schedules them for transmission over the available radio spectrum, and in the UE-to-Internet direction it assembles physical layer segments into IP packets and forwards them (over a GTP/UDP/IP tunnel) to the upstream user plane of the Mobile Core. Also, based on observations of the wireless channel quality and per-subscriber policies, it decides whether to

(a) forward outgoing packets directly to the UE, (b) indirectly forward packets to the UE via a neighboring base station, or (c) utilize multiple paths to reach the UE. The third case has the option of either spreading the physical payloads across multiple base stations or across multiple carrier frequencies of a single base station (including Wi-Fi).

Note that as outlined in the previous section, scheduling is complex and multi-faceted, even when viewed as a localized decision at a single base station. What we now see is that there is also a global element, whereby it's possible to forward traffic to a different base station (or to multiple base stations) in an effort to make efficient use of the radio spectrum over a larger geographic area.

In other words, the RAN as a whole (i.e., not just a single base station) not only supports handovers (an obvious requirement for mobility), but also *link aggregation* and *load balancing*, mechanisms that are familiar to anyone that understands the Internet. We will revisit how such RAN-wide (global) decisions can be made using SDN techniques in a later section.

Mobile Core

The main function of the Mobile Core is to provide external packet data network (e.g., Internet) connectivity to mobile subscribers, while ensuring that they are authenticated and their observed service qualities satisfy their subscription SLAs. An important aspect of the Mobile Core is that it needs to manage all subscribers' mobility by keeping track of their last whereabouts at the granularity of the serving base station.

While the aggregate functionality remains largely the same as we migrate from 4G to 5G, how that functionality is virtualized and factored into individual components changes, with the 5G Mobile Core heavily influenced by the cloud's march towards a microservice-based (cloud native) architecture. This shift to cloud native is deeper than it might first appear, in part because it opens the door to customization and specialization. Instead of supporting just voice and broadband connectivity, the 5G Mobile Core can evolve to also support, for example, massive IoT, which has a fundamentally different latency requirement and usage pattern (e.g., many more devices connecting intermittently). This stresses—if not breaks—a one-size-fits-all approach to session management.

4G Mobile Core

The 4G Mobile Core, which 3GPP officially refers to as the *Evolved Packet Core (EPC)*, consists of five main components, the first three of which run in the Control Plane (CP) and the second two of which run in the User Plane (UP):

- MME (Mobility Management Entity): Tracks and manages the movement of UEs throughout the RAN. This includes recording when the UE is not active.
- HSS (Home Subscriber Server): A database that contains all subscriber-related information.
- PCRF (Policy & Charging Rules Function): Tracks and manages policy rules and records billing data on subscriber traffic.
- SGW (Serving Gateway): Forwards IP packets to and from the RAN. Anchors the Mobile Core end of the bearer service to a (potentially mobile) UE, and so is involved in handovers from one base station to another.
- PGW (Packet Gateway): Essentially an IP router, connecting the Mobile Core to the external Internet. Supports additional access-related functions, including policy enforcement, traffic shaping, and charging.

Although specified as distinct components, in practice the SGW (RAN-facing) and PGW (Internet-facing) are often combined in a single device, commonly referred to as an S/PGW. The end result is illustrated in Figure 3.9.

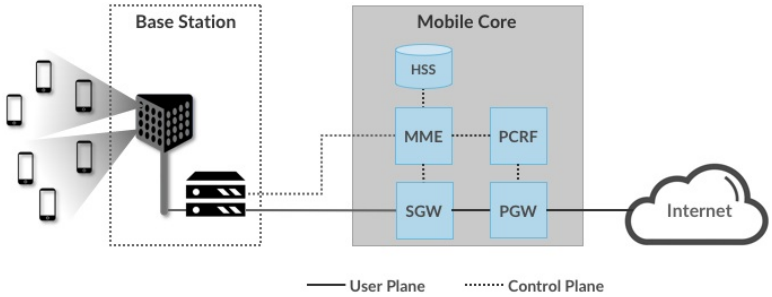

Figure 3.9: 4G Mobile Core (Evolved Packet Core).

Note that 3GPP is flexible in how the Mobile Core components are deployed to serve a geographic area. For example, a single MME/PGW pair might serve a metropolitan area, with SGWs deployed across ~10 edge sites spread throughout the city, each of which serves ~100 base stations. But alternative deployment configurations are allowed by the spec.

5G Mobile Core

The 5G Mobile Core, which 3GPP calls the *NG-Core*, adopts a microservice-like architecture, where we say "microservice-like" because while the 3GPP specification spells out this level of disaggregation, it is really just prescribing a set of functional blocks and not an implementation. Keeping in mind a set of functional blocks is very different from the collection of engineering decisions that go into designing a microservice-based system, viewing the collection of components shown in Figure 3.10 as a set of microservices is a good working model.

The following organizes the set of functional blocks into three groups. The first group runs in the Control Plane (CP) and has a counterpart in the EPC:

- AMF (Core Access and Mobility Management Function): Manages the mobility-related aspects of the EPC's MME. Responsible for connection and reachability management, mobility management, access authentication and authorization, and location services.
- SMF (Session Management Function): Manages each UE session, including IP address allocation, selection of associated UP function, control aspects of QoS, and control aspects of UP routing. Roughly corresponds to part of the EPC's MME and the control-related aspects of the EPC's PGW.
- PCF (Policy Control Function): Manages the policy rules that other CP functions then enforce. Roughly corresponds to the EPC's PCRF.
- UDM (Unified Data Management): Manages user identity, including the generation of authentication credentials. Includes part of the functionality in the EPC's HSS.
- AUSF (Authentication Server Function): Essentially an authentication server. Includes part of the functionality in the EPC's HSS.

The second group also runs in the Control Plane (CP) but does not have a counterpart in the EPC:

- SDSF (Structured Data Storage Network Function): A "helper" service used to store structured data. Might be implemented by an "SQL Database" in a microservices-based system.
- UDSF (Unstructured Data Storage Network Function): A "helper" service used to store unstructured data. Might be implemented by a "Key/Value Store" in a microservices-based system.
- NEF (Network Exposure Function): A means to expose select capabilities to third-party services, including translation between internal and external representations for data. Might be implemented by an "API Server" in a microservices-based system.

- NFR (NF Repository Function): A means to discover available services. Might be implemented by a “Discovery Service” in a microservices-based system.
- NSSF (Network Slicing Selector Function): A means to select a Network Slice to serve a given UE. Network slices are essentially a way to differentiate service given to different users. It is a key feature of 5G that we discuss in depth later in this tutorial.

The third group includes the one component that runs in the User Plane (UP):

- UPF (User Plane Function): Forwards traffic between RAN and the Internet, corresponding to the S/PGW combination in EPC. In addition to packet forwarding, responsible for policy enforcement, lawful intercept, traffic usage reporting, and QoS policing.

Of these, the first and third groups are best viewed as a straightforward refactoring of 4G’s EPC, while the second group—despite the gratuitous introduction of new terminology—is 3GPP’s way of pointing to a cloud native solution as the desired end-state for the Mobile Core. Of particular note, introducing distinct storage services means that all the other services can be stateless, and hence, more readily scalable. Also note that Figure 3.10 adopts an idea that’s common in microservice-based systems, namely, to show a “message bus” interconnecting all the components rather than including a full set of pairwise connections. This also suggests a well-understood implementation strategy.

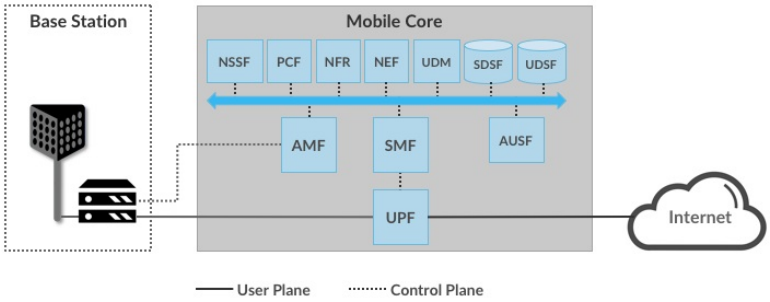

FIgure 3.10: 5G Mobile Core (NG-Core).

Stepping back from these details, and with the caveat that we are presuming an implementation, the main takeaway is that we can conceptualize the Mobile Core as a *Service Mesh*. We adopt this terminology for “an interconnected set of microservices” since it is widely used in cloud native systems. Other terms you will sometimes hear are *Service Graph* and *Service Chain*, the latter being more prevalent in NFV-oriented documents. 3GPP is silent on the specific terminology since it is considered an implementation choice rather than part of the specification.

Deployment Options

With an already deployed 4G RAN/EPC in the field and a new 5G RAN/NG-Core deployment underway, we can’t ignore the issue of transitioning from 4G to 5G (an issue the IP-world has been grappling with for 20 years). 3GPP officially spells out multiple deployment options, which can be summarized as follows:

- Stand-Alone 4G / Stand-Alone 5G
- Non-Stand-Alone (4G+5G RAN) over 4G’s EPC
- Non-Stand-Alone (4G+5G RAN) over 5G’s NG-Core

Focusing on the second pair, which imply incremental phasing, we see two general strategies. The first is to connect new 5G base stations to existing 4G-based EPCs, and then incrementally evolve the Mobile Core by refactoring the components and adding NG-Core capabilities over time. The second is to implement a backward-compatible NG-Core that can support both 4G and 5G base stations, where the new NG-Core could be implemented from scratch, but would likely start with the existing EPC code base.

One reason we call attention to the phasing issue is that we face a similar challenge in the sections that follow. The closer the following discussion gets to implementation details, the more specific we have to be about whether we are using 4G components or 5G components. As a general rule, we use 4G components—particularly with respect to the Mobile Core, since that's the available open source software—and trust the reader can make the appropriate substitution without loss of generality. Like the broader industry, the open source community is in the process of incrementally evolving its 4G code base into its 5G-compliant counterpart.

Chapter 4: RAN Internals

The description of the RAN in the previous chapter focused on functionality, but was mostly silent about the RAN's internals structure. We now focus in on some of the internal details, and in doing so, explain how the RAN is being transformed in 5G. This involves first describing the stages in the packet processing pipeline, and then showing how these stages can be distributed and implemented.

Packet Processing Pipeline

Figure 4.1 shows the packet processing stages implemented by the base station. These stages are specified by the 3GPP standard. Note that the figure depicts the base station as a pipeline (running left-to-right) but it is equally valid to view it as a protocol stack (as is typically done in official 3GPP documents). Also note that (for now) we are agnostic as to how these stages are implemented, but since we are ultimately heading towards a cloud-based implementation, you can think of each as corresponding to a microservice (if that is helpful).

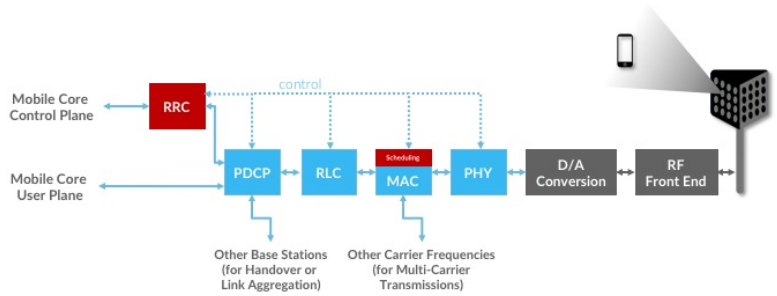

Figure 4.1: RAN processing pipeline, including both user and control plane components.

The key stages are as follows:

- RRC (Radio Resource Control) → Responsible for configuring the coarse-grain and policy-related aspects of the pipeline. The RRC runs in the RAN's control plane; it does not process packets on the user plane.
- PDCP (Packet Data Convergence Protocol) → Responsible for compressing and decompressing IP headers, ciphering and integrity protection, and making an "early" forwarding decision (i.e., whether to send the packet down the pipeline to the UE or forward it to another base station).
- RLC (Radio Link Control) → Responsible for segmentation and reassembly, including reliably transmitting/receiving segments by implementing ARQ.
- MAC (Media Access Control) → Responsible for buffering, multiplexing and demultiplexing segments, including all real-time scheduling decisions about what segments are transmitted when. Also able to make a "late" forwarding decision (i.e., to alternative carrier frequencies, including Wi-Fi).
- PHY (Physical Layer) → Responsible for coding and modulation (as discussed in an earlier chapter), including FEC.

The last two stages in Figure 4.1 (D/A conversion and the RF front-end) are beyond the scope of this book.

While it is simplest to view the stages in Figure 4.1 as a pure left-to-right pipeline, in practice the Scheduler running in the MAC stage implements the "main loop" for outbound traffic, reading data from the upstream RLC and scheduling transmissions to the downstream PHY. In particular, since the Scheduler determines the number of bytes to transmit

to a given UE during each time period (based on all the factors outlined in an earlier chapter), it must request (get) a segment of that length from the upstream queue. In practice, the size of the segment that can be transmitted on behalf of a single UE during a single scheduling interval can range from a few bytes to an entire IP packet.

Split RAN

The next step is understanding how the functionality outlined above is partitioned between physical elements, and hence, "split" across centralized and distributed locations. Although the 3GPP standard allows for multiple split-points, the partition shown in Figure 4.2 is the one being actively pursued by the operator-led O-RAN (Open RAN) Alliance. It is the split we adopt throughout the rest of this book.

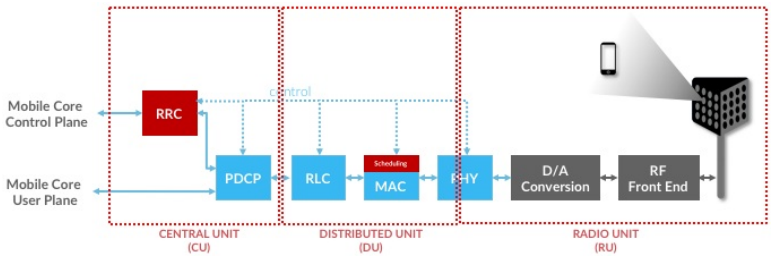

Figure 4.2: Split-RAN processing pipeline distributed across a Central Unit (CU), Distributed Unit (DU), and Radio Unit (RU).

This results in a RAN-wide configuration similar to that shown in Figure 4.3, where a single *Central Unit (CU)* running in the cloud serves multiple *Distributed Units (DUs)*, each of which in turn serves multiple *Radio Units (RUs)*. Critically, the RRC (centralized in the CU) is responsible for only near-real time configuration and control decision making, while the Scheduler that is part of the MAC stage is responsible for all real-time scheduling decisions.

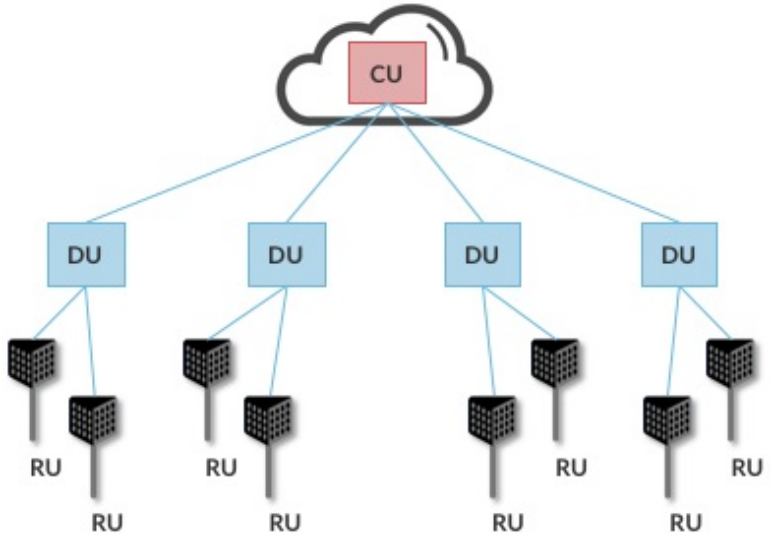

Figure 4.3: Split-RAN hierarchy, with one CU serving multiple DUs, each of which serves multiple RUs.

Clearly, a DU needs to be "near" (within 1ms) the RUs it manages since the MAC schedules the radio in real-time. One familiar configuration is to co-locate a DU and an RU in a cell tower. But when an RU corresponds to a small cell, many of which might be spread across a modestly sized geographic area (e.g., a mall, campus, or factory), then a single DU would likely service multiple RUs. The use of mmWave in 5G is likely to make this later configuration all the more common.

Also note that the split-RAN changes the nature of the Backhaul Network, which in 4G connected the base stations (eNBs) back to the Mobile Core. With the split-RAN there are multiple connections, which are officially labelled as follows:

- RU-DU connectivity is called the Fronthaul
- DU-CU connectivity is called the Midhaul
- CU-Mobile Core connectivity is called the Backhaul

As we will see in a later chapter, one possible deployment co-locates the CU and Mobile Core in the same cluster, meaning the backhaul is implemented in the cluster switching fabric. In such a configuration, the midhaul then effectively serves the same purpose as the original backhaul, and the fronthaul is constrained by the predictable/low-latency requirements of the MAC stage's real-time scheduler.

Software-Defined RAN

Finally, we describe how the RAN is implemented according to SDN principles, resulting in an SD-RAN. The key architectural insight is shown in Figure 4.4, where the RRC from Figure 4.1 is partitioned into two sub-components: the one on the left provides a 3GPP-compliant way for the RAN to interface to the Mobile Core's control plane, while the latter opens a new programmatic API for exerting software-based control over the pipeline that implements the RAN user plane. To be more specific, the left sub-component simply forwards control packets between the Mobile Core and the PDCP, providing a path over which the Mobile Core can communication with the UE for control purposes, whereas the right sub-component implements the core of the RCC's control functionality.

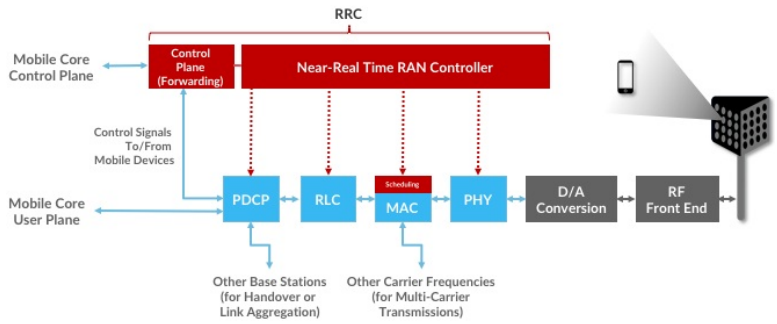

Figure 4.4: RRC disaggregated into a Mobile Core facing control plane component and a Near Real-Time Controller.

Although not shown in Figure 4.4, keep in mind (from Figure 4.2) that all constituent parts of the RRC, plus the PDCP, form the CU.

Completing the picture, Figure 4.5 shows the Near-RT RAN Controller implemented as a traditional SDN Controller hosting a set of SDN control apps. The Near Real-Time Controller maintains a RAN Network Information Base (R-NIB) that includes time-averaged CQI values and other per-session state (e.g., GTP tunnel IDs, QCI values for the type of traffic), while the MAC (as part of the DU) maintains the instantaneous CQI values required by the real-time scheduler. Specifically, the R-NIB includes the following state:

- NODES: Base Stations and Mobile Devices
  - Base Station Attributes:
    - Identifiers
    - Version
    - Config Report
    - RRM config
    - PHY resource usage
  - Mobile Device Attributes:
    - Identifiers

        - Capability
        - Measurement Config
        - State (Active/Idle)
- LINKS: Physical between two nodes and potential between UEs and all neighbor cells
    - Link Attributes:
        - Identifiers
        - Link Type
        - Config / Bearer Parameters
        - QCI Value
- SLICES: Virtualized RAN construct
    - Slice Attributes:
        - Links
        - Bearers / Flows
        - Validity Period
        - Desired KPIs
        - MAC RRM Configuration
        - RRM Control Configuration

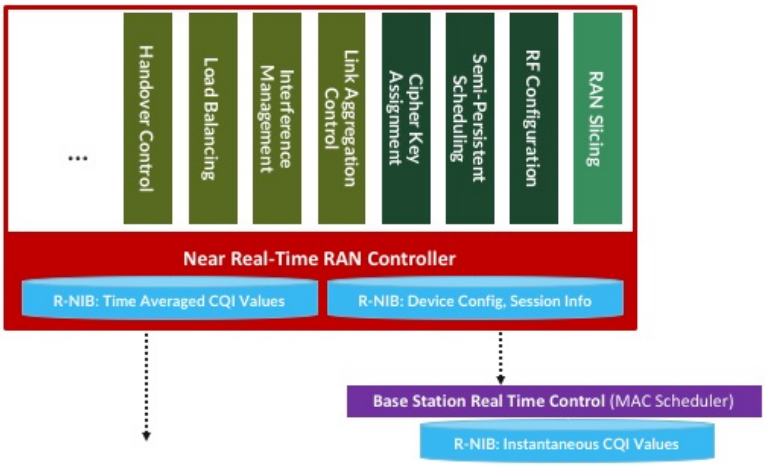

FIgure 4.5: Example set of control applications running on top of Near Real-Time RAN Controller.

The example Control Apps in Figure 4.5 include a range of possibilities, but is not intended to be an exhaustive list. The right-most example, RAN Slicing, is the most ambitious in that it introduces a new capability: Virtualizing the RAN. It is also an idea that has been implemented, which we describe in more detail in the next chapter.

The next three (RF Configuration, Semi-Persistent Scheduling, Cipher Key Assignment) are examples of configuration-oriented applications. They provide a programmatic way to manage seldom-changing configuration state, thereby enabling zero-touch operations. Coming up with meaningful policies (perhaps driven by analytics) is likely to be an avenue for innovation in the future.

Chapter 5: Advanced Capabilities

Disaggregating the cellular network pays dividends. This section explores three examples. Stepping back to look at the big picture, the Architecture section described “what is” (the basics of 3GPP) and the RAN Internals section described “what will be” (where the industry, led by the O-RAN Alliance, is clearly headed), whereas this section describes “what could be” (our best judgement on cutting-edge capabilities that will eventually be realized).

Optimized Data Plane

There are many reasons to disaggregate functionality, but one of the most compelling is that by decoupling control and data code paths, it is possible to optimize the data path. This can be done, for example, by programming it into specialized hardware. Modern white-box switches with programmable packet forwarding pipelines are a perfect example of specialized hardware we can exploit in the cellular network. Figure 5.1 shows the first step in the process of doing this. The figure also pulls together all the elements we’ve been describing up to this point. There are several things to note about this diagram.

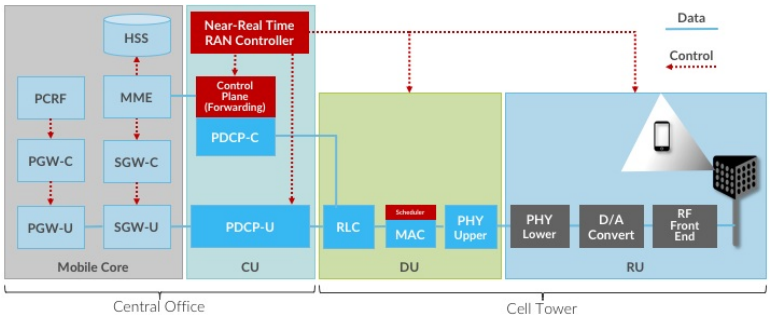

Figure 5.1: End-to-end disaggregated system, including Mobile Core and Split-RAN.

First, the figure combines both the Mobile Core and RAN elements, organized according to the major subsystems: Mobile Core, Central Unit (CU), Distributed Unit (DU), and Radio Unit (RU). The figure also shows one possible mapping of these subsystems onto physical locations, with the first two co-located in a Central Office and the latter two co-located in a cell tower. As discussed earlier, other configurations are also possible.

Second, the figure shows the Mobile Core’s two user plane elements (PGW, SGW) and the Central Unit’s single user plane element (PDCP) further disaggregated into control/user plane pairs, denoted PGW-C / PGW-U, SGW-C / SGW-U, and PDCP-C / PDCP-U, respectively. Exactly how this decoupling is realized is a design choice (i.e., not specified by 3GPP), but the idea is to reduce User Plane component to the minimal Receive-Packet / Process-Packet / Send-Packet processing core, and elevate all control-related aspects into the Control Plane component.

Third, the PHY (Physical) element of the RAN pipeline is split between the DU and RU partition. Although beyond the scope of this book, the 3GPP spec specifies the PHY element as a collection of functional blocks, some of which can be effectively implemented by software running on a general-purpose processor and some of which are best implemented in specialized hardware (e.g., a Digital Signal Processor). These two subsets of functional blocks map to the PHY Upper (part of the DU) and the PHY Lower (part of the RU), respectively.

Fourth, and somewhat confusingly, Figure 5.1 shows the PCDP-C element and the Control Plane (Forwarding) element combined into a single functional block, with a data path (blue line) connecting that block to both the RLC and the MME. Exactly how this pair is realized is an implementation choice (e.g., they could map onto two or more

microservices), but the end result is that they are part of an end-to-end path over which the MME can send control packets to the UE. Note that this means responsibility for demultiplexing incoming packets between the control plane and user plane falls to the RLC.

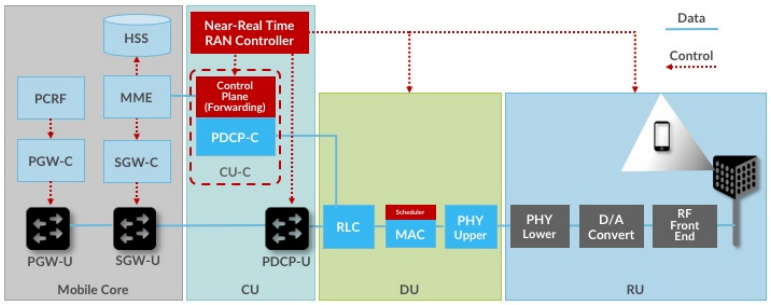

FIgure 5.2: Implementing data plane elements of the User Plane in programmable switches.

Figure 5.2 shows why we disaggregated these components: it allows us to realize the three user plane elements (PGW-U, SGW-U, PDCP-U) in switching hardware. As we will expand on in more detail in a later section, this can be done using a combination of a language that is tailored for programming forwarding pipelines (e.g., P4), and a protocol-independent switching architecture (e.g., Tofino). For now, the important takeaway is that the RAN and Mobile Core user plane can be mapped directly onto an SDN-enabled data plane.

Pushing RAN and Mobile Core forwarding functionality into the switching hardware results in overlapping terminology that can sometimes be confusing. 5G separates the functional blocks into control and user planes, while SDN disaggregates a given functional block into control and data plane halves. The overlap comes from our choosing to implement the 5G user plane by splitting it into its SDN-based control and data plane parts. As one simplification, we refer to the Control Plane (Forwarding) and PDCP-C combination as the CU-C (Central Unit - Control) going forward.

Finally, the SDN-prescribed control/data plane disaggregation comes with an implied implementation strategy, namely, the use of a scalable and highly available *Network Operating System (NOS)*. Like a traditional OS, a NOS sits between application programs (control apps) and the underlying hardware devices (whitebox switches), providing higher levels abstractions (e.g., network graph) to those applications, while hiding the low-level details of the underlying hardware. To make the discussion more concrete, we use ONOS (Open Network Operating System) as an example NOS, where PGW-C, SGW-C, and PDCP-C are all realized as control applications running on top of ONOS.

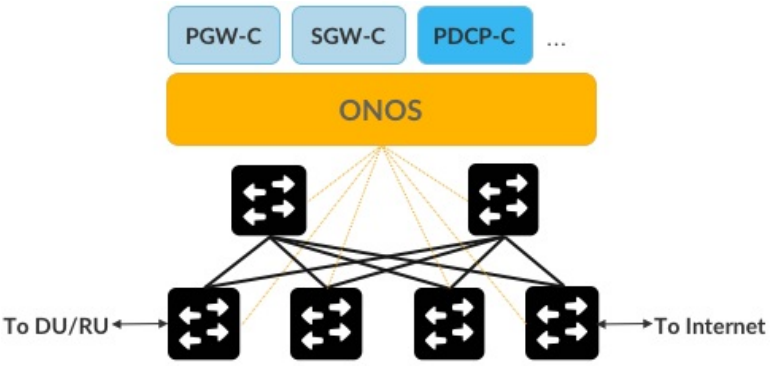

FIgure 5.3: Control Plane elements of the User Plane implemented as Control Applications running on an SDN Controller (e.g., ONOS).

Figure 5.3 shows one possible configuration, in which the underlying switches are interconnected to form a leaf-spine fabric. Keep in mind that the linear sequence of switches implied by Figure 5.2 is logical, but that in no way restricts the actual hardware to the same topology. The reason we use a leaf-spine topology is related to our ultimate goal of building an edge cloud, and leaf-spine is the proto-typical structure for such cloud-based clusters. This means the three control applications must work in concert to implement an end-to-end path through the fabric, which in practice

happens with the aid of other, fabric aware, control applications (as implied by the "..." in the Figure). We describe the complete picture in more detail in a later section, but for now, the big takeaway is that the control plane components of the 5G overlay can be realized as control applications for an SDN-based underlay.

Multi-Cloud

Another consequence of disaggregating functionality is that once decoupled, different functions can be placed in different physical locations. We have already seen this when we split the RAN, placing some functions (e.g., the PCDP and RRC) in the Central Unit and others (e.g., RLC and MAC) in Distributed Units. This allows for simpler (less expensive) hardware in remote locations, where there are often space, power, and cooling constraints.

This process can be repeated by distributing the more centralized elements across multiple clouds, including large datacenters that already benefit from elasticity and economies of scale. Figure 5.4 shows the resulting multi-cloud realization of the Mobile Core. We leave the user plane at the edge of the network (e.g., in the Central Office) and move control plane to a centralized cloud. It could even be a public cloud like Google or Amazon. This includes not only the MME, PCRF and HSS, but also the PGW-C and SGW-C we decoupled in the previous section. (Note that Figure 5.4 renames the PDCP-U from earlier diagrams as the CU-U; either label is valid.)

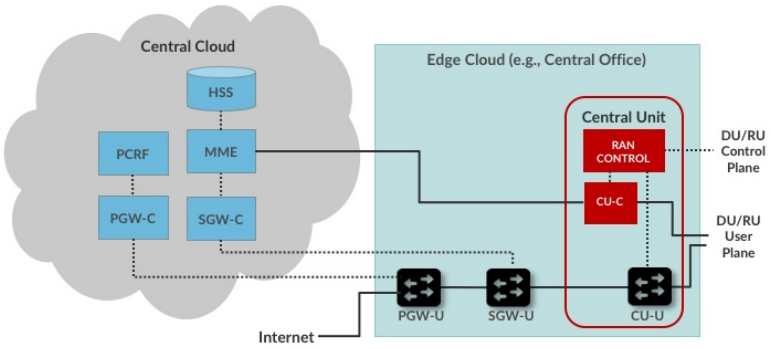

FIgure 5.4: Multi-Cloud implementation, with MME, HSS, PCRF and Control Plane elements of the PGW and SGW running in a centralized cloud.

What is the value in doing this? Just like the DU and RU, the Edge Cloud likely has limited resources. If we want room to run new edge services there, it helps to move any components that need not be local to a larger facility with more abundant resources. Centralization also facilitates collecting and analyzing data across multiple edge locations, which is harder to do if that information is distributed over multiple sites. (Analytics performed on this data also benefits from having abundant compute resources available.)

But there's another reason worth calling out: It lowers the barrier for anyone (not just the companies that own and operate the RAN infrastructure) to offer mobile services to customers. These entities are called *MVNOs (Mobile Virtual Network Operators)* and one clean way to engineer an MVNO is to run your own Mobile Core on a cloud of your choosing.

Network Slicing

One of the most compelling value propositions of 5G is the ability to differentiate the level of service offered to different applications and customers. Differentiation, of course, is key to being able to charge some customers more than others, but the monetization case aside, it is also necessary if you are going to support such widely varying applications as streaming video (which requires high bandwidth but can tolerate larger latencies) and Internet-of-Things (which has minimal bandwidth needs but requires extremely low and predictable latencies).

The mechanism that supports this sort of differentiation is called network slicing, and it fundamentally comes down to scheduling, both in the RAN (deciding which segments to transmit) and in the Mobile Core (scaling microservice instances and placing those instances on the available servers). The following introduces the basic idea, starting with the RAN.

But before getting into the details, we note that a network slice is a generalization of the QoS Class Index (QCI) discussed earlier. 3GPP specifies a standard set of network slices, called *Standardized Slice Type (SST)* values. For example, SST 1 corresponds to mobile broadband, SST 2 corresponds to Ultra-Reliable Low Latency Communications, SST 3 corresponds to Massive IoT, and so on. It is also possible to extend this standard set with additional slice behaviors, as well as define multiple slices for each SST (e.g., to further differentiate subscribers based on priority).

RAN Slicing

We start by reviewing the basic scheduling challenge previewed in Section 2. As depicted in Figure 5.5, the radio spectrum can be conceptualized as a two-dimensional grid of *Resource Blocks (RB)*, where the scheduler's job is to decide how to fill the grid with the available segments from each user's transmission queue based on CQI feedback from the UEs. To restate, the power of OFDMA is the flexibility it provides in how this mapping is performed.

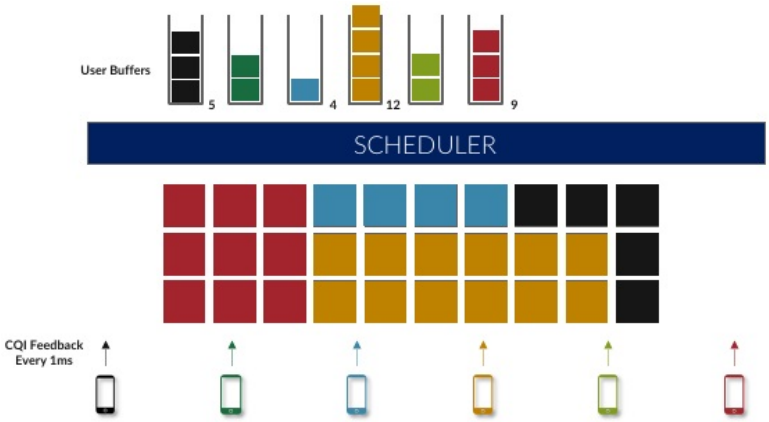

Figure 5.5: Scheduler allocating resource blocks to UEs.

While in principle one could define an uber scheduler that takes dozens of different factors into account, the key to network slicing is to add a layer of indirection, such that (as shown in Figure 5.6), we perform a second mapping of Virtual RBs to Physical RBs. This sort of virtualization is common in resource allocators throughout computing systems because we want to separate how many resources are allocated to each user from the decision as to which physical resources are actually assigned. This virtual-to-physical mapping is performed by a layer typically known as a *Hypervisor*, and the important thing to keep in mind is that it is totally agnostic about which user's segment is affected by each translation.

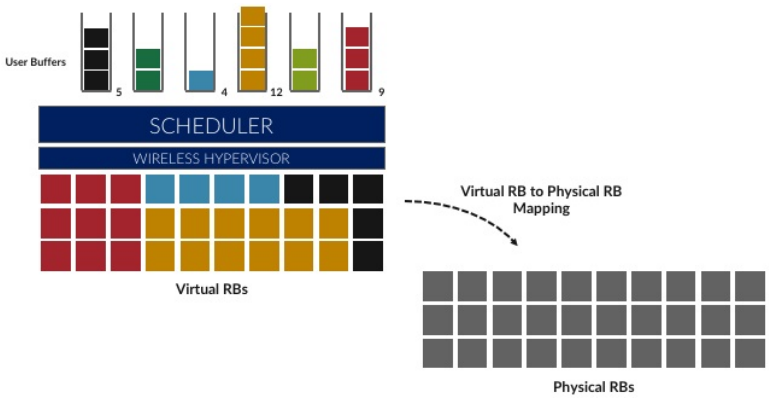

FIgure 5.6: Wireless Hypervisor mapping virtual resource blocks to physical

resource blocks

Having decoupled the Virtual RBs from Physical RBs, it is now possible to define multiple Virtual RB sets (of varying sizes), each with its own scheduler. Figure 5.7 gives an example with two equal-sized RB sets, where the important consequence is that having made the macro-decision that the Physical RBs are divided into two equal partitions, the scheduler associated with each partition is free to allocate Virtual RBs completely independent from each other. For example, one scheduler might be designed to deal with high-bandwidth video traffic and another scheduler might be optimized for low-latency IoT traffic. Alternatively, a certain fraction of the available capacity could be reserved for premium customers or other high-priority traffic (e.g., public safety), with the rest shared among everyone else.

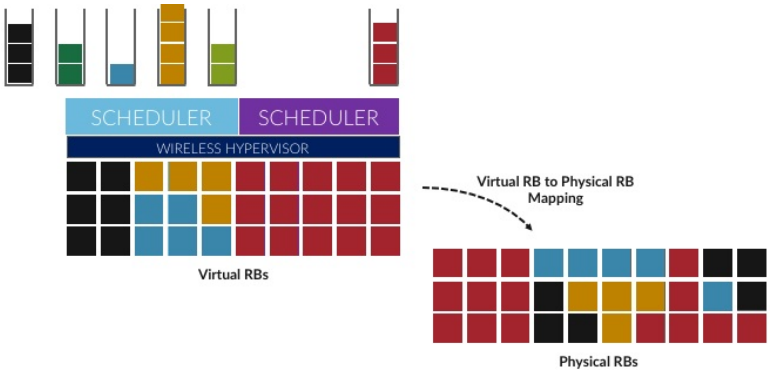

FIgure 5.7: Multiple schedulers running on top of wireless hypervisor.

Going one level deeper in the implementation details, the real-time scheduler running in each DU receives high-level directives from the near real-time scheduler running in the CU, and as depicted in Figure 5.8, these directives make dual transmission, handoff, and interference decisions on a per-slice basis. A single RAN Slicing control application is responsible for the macro-scheduling decision by allocating resources among a set of slices. Understanding this implementation detail is important because all of these control decisions are implemented by software modules, and hence, easily changed or customized. They are not “locked” into the underlying system, as they have historically been in 4G’s eNodeBs.

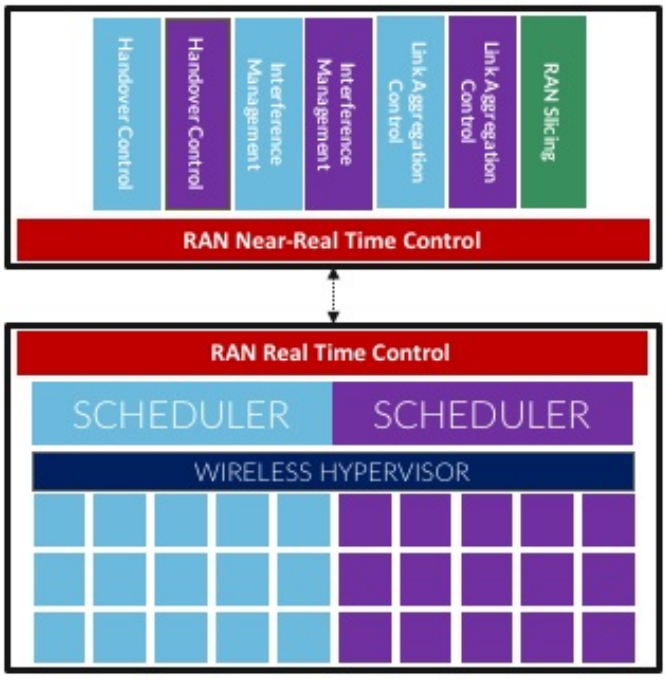

FIgure 5.8: Centralized near-realtime control applications cooperating with distribute real-time RAN schedulers.

Core Slicing

In addition to slicing the RAN, we also need to slice the Mobile Core. In many ways, this is a well-understood problem, involving QoS mechanisms in the network switches (i.e., making sure packets flow through the switching fabric according to the bandwidth allocated to each slice) and the cluster processors (i.e., making sure the containers that implement each microservice are allocated sufficient CPU cores to sustain the packet forwarding rate of the corresponding slice).

But packet scheduling and CPU scheduling are low-level mechanisms. What makes slicing work is to also virtualize and replicate the entire service mesh that implements the Mobile Core. If you think of a slice as a system abstraction, then that abstraction needs to keep track of the set of interconnected set of microservices that implement each slice, and then instruct the underlying packet schedulers to allocate sufficient network bandwidth to the slice's flows and the underlying CPU schedulers to allocate sufficient compute cycles to the slice's containers.

For example, if there are two network slices (analogous to the two RAN schedulers shown in Figures 5.7 and 5.8), then there would also need to be two Mobile Core service meshes: One set of AMF, SMF, UPF,... microservices running on behalf of the first slice and a second set of AMF, SMF, UPF,... microservices running on behalf of the second slice. These two meshes would scale independently (i.e., include a different number of container instances), depending on their respective workloads and QoS guarantees. The two slices would also be free to make different implementation choices, for example, with one optimized for massive IoT applications and the other optimized for high-bandwidth AR/VR applications.

The one remaining mechanism we need is a demultiplexing function that maps a given packet flow (e.g., between UE and some Internet application) onto the appropriate instance of the service mesh. This is the job of the NSSF described in an earlier section: it is responsible for selecting the mesh instance a given slice’s traffic is to traverse.

Chapter 6: Exemplar Implementation

The steps we've taken in the previous chapters to virtualize, disaggregate, optimize, distribute, and slice the cellular network not only help us understand the inner-workings of 5G, but they are also necessary to reduce the entirety of the 5G cellular network to practice. The goal is an implementation, which by definition, forces us to make specific engineering choices. This chapter describes one set of engineering choices that results in a running system. It should be interpreted as an exemplar, for the sake of completeness, but not the only possibility.

The system we describe is called CORD, which you will recall from the Introduction is an acronym (**C**entral **O**ffice **R**e-architected as a **D**atacenter). More concretely, CORD is a blueprint for building a 5G deployment from commodity hardware and a collection of open source software components. We call this hardware/software combination a CORD POD, where the idea is to deploy a POD at each edge site that is part of a cellular network. The following describes CORD in terms of a set of engineering decisions. It is not a substitute for detailed documentation for installing, developing, and operating CORD, which can be found elsewhere: https://guide.opencord.org.

> As discussed in the Introduction, even though CORD includes "Central Office" in its name, a CORD POD is a general design, and not strictly limited to being deployed in a conventional Central Office.

Before getting into the specifics, it is important to understand that CORD is a work-in-progress, with a sizable open source community contributing to its code base. Many of the components are quite mature, and currently running in operator trials and production networks. Others (largely corresponding to the advanced capabilities described in the previous chapter) are prototypes that run in "demonstration mode," but are not yet complete enough to be included in official releases. Also, as outlined in the earlier discussion on deployment options, CORD starts with a production-quality EPC that is being incrementally evolved into its 5G counterpart. (This chapter uses the EPC-specific components for illustrative purposes.)

Framework

Figure 6.1 gives a schematic overview of a CORD POD. It connects downstream to a set of DUs (and associated RUs), and upstream to the rest of the Internet. Internally, it includes a set of commodity servers (the figure shows four racks of three servers each, but the design accommodates anywhere from a partial rack to 16 racks) and a set of white-box switches arranged in a leaf-spine topology (the figure shows two leaves and two spine switches, but the design allows anywhere from a single switch to two leaf switches per rack and as many spine switches as necessary to provide sufficient east-to-west bandwidth).

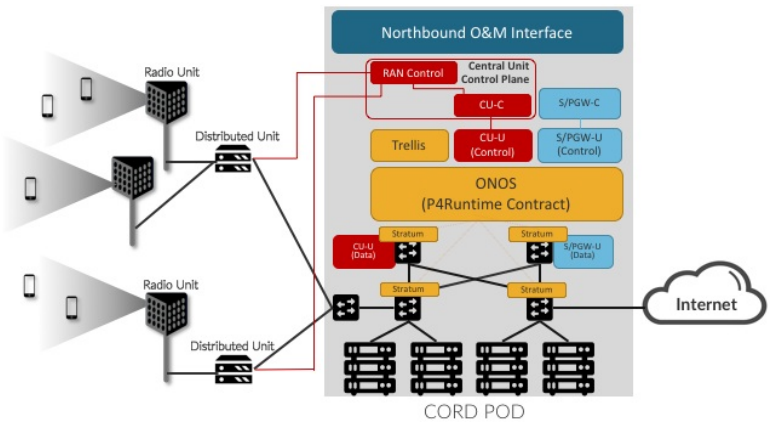

Figure 6.1: CORD implementation of RAN and Mobile Core.

With respect to software, Figure 6.1 shows a combination of RAN (red) and Mobile Core (blue) components, plus the modules that define the CORD platform (orange). We describe the platform components later in this chapter, but you can think of them as collectively implementing a multi-tenant cloud on top of which many different scalable services can run. The RAN and Mobile Core are two such tenants. The CORD platform can also host other edge services (which is one reason CORD is built using cloud technology in the first place), but exactly what other edge services might run on a given CORD POD is beyond the scope of this book.

The RAN and Core related components are ones we've described in earlier chapters. They include the Control and User planes of the CU and Mobile Core, respectively, where to simplify the diagram, we show the SGW and PGW merged into a single S/PGW. One aspect of Figure E that requires further elaboration is how each of the RAN and Mobile Core components are actually realized. There are three different manifestations of the functional components implied by the Figure: (1) the data plane layer of the CU-U and S/PGW-U are realized as P4 programs loaded into the programmable switches; (2) the control plane layer of the CU-U and S/PGW-U (as well as the Trellis platform module) are realized as control applications loaded onto the ONOS Network OS; and the remaining components are realized as Kubernetes-managed microservices. (Kubernetes is implicit, and not shown in the figure.)

To expand on this idea, Figure 6.2 gives an alternative view of a CORD POD, abstracting away the details of *what* services it hosts, and focusing instead on *how* those services are instantiated on the POD. In this figure, all the functionality instantiated onto the POD runs as a combination of Kubernetes-based microservices and ONOS-based control applications.

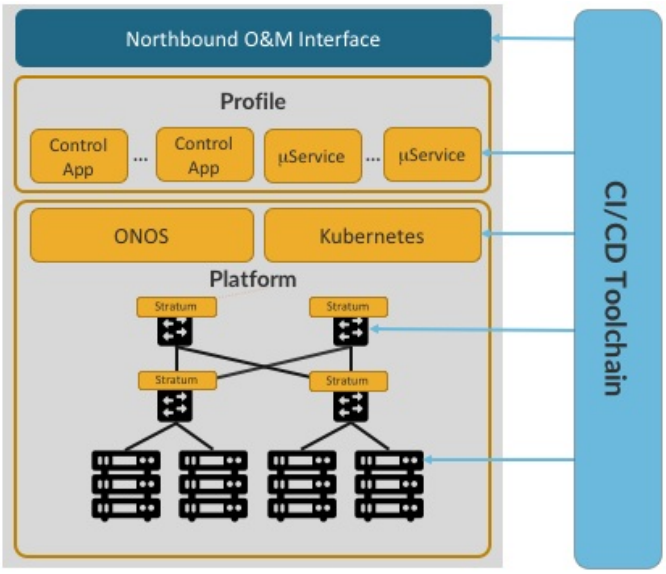

Figure 6.2: Alternative view of CORD, with a CI/CD toolchain managing the platform and set of services implemented by a combination of ONOS-based control apps and Kubernetes-based microservices.

When abstracted in this way, we can view a POD as including three major subsystems:

- **Platform:** The base layer common to all configurations includes Kubernetes as the container management system and ONOS as the SDN controller, with Stratum loaded on to each switch. Both ONOS and the control applications it hosts run as a Kubernetes-managed microservices.
- **Profile:** The deployment-specific collection of microservices and SDN control apps that have been selected to run on a particular POD. This is a variable and evolvable set, and it includes the control plane and edge services described elsewhere.
- **CI/CD Toolchain:** Used to assemble, deploy, operate, and upgrade a particular Platform/Profile combination. It implements a set of processes that transforms a collection of disaggregated and virtualized components into an operational system capable of responding to operator directives and carrying live traffic.

Although beyond the scope of this book, the CI/CD toolchain uses standard DevOps tools to bootstrap software onto the cluster of servers and switches, and rollout/rollback individual microservices and control applications. It also auto-generates the Northbound Interface (NBI) that operators use to manage the POD, based on a declarative specification of the Profile the POD is configured to support. This NBI is sufficiently complete to operate a CORD POD in a production environment.

Platform Components

We now return to the three platform-related components shown in Figures 6.1 and 6.2. Each is a substantial open source component in its own right, but for our purposes, it is enough to understand the role each plays in supporting a 5G-based profile of CORD.

- **Stratum:** A thin operating system that runs locally on each white-box switch. It's purpose is to provide a hardware-independent interface for managing and programming the switches in CORD. This includes using *P4* to define the forwarding behavior of the switch's forwarding pipeline (think of this program as a contract between the control and data planes), and *P4Runtime* to control that forwarding contract at runtime.
- **ONOS:** A Network Operating System used to configure and control a network of programmable white-box switches. It runs off-switch as a logically centralized SDN controller, and hosts a collection of SDN control applications, each of which controls some aspect of the underlying network. Because it is logically centralized, ONOS is designed to be highly available and to have scalable performance.
- **Trellis:** An ONOS-hosted SDN control application that implements a leaf-spine fabric on a network of white-box switches. It implements the control plane for several features, including VLANs and L2 bridging, IPv4 and IPv6 unicast and multicast routing, DHCP L3 relay, dual-homing of servers and upstream routers, QinQ forwarding/termination, MPLS pseudowires, and so on. In addition, Trellis can make the entire fabric appear as a single (virtual) router to upstream routers, which communicate with Trellis using standard BGP.

Stratum (running on each switch) and ONOS (running off-switch and managing a network of switches) communicate using the following interfaces:

- **P4:** Defines the forwarding behavior for programmable switching chips as well as model fixed-function ASIC pipelines. A P4 program defines a contract that is implemented by the data plane and programmable by the control plane.
- **P4Runtime:** An SDN-ready interface for controlling forwarding behavior at runtime. It is the key for populating forwarding tables and manipulating forwarding state, and it does so in a P4 program and hardware agnostic way.
- **OpenConfig Models:** Define a base for device configuration and management. These models can be programmatically extended for platform-specific functionality, but the goal is to minimize model deviations so as to enable a vendor-agnostic management plane.
- **gNMI** (gRPC Network Management Interface): Improves on the existing configuration interfaces by using a binary representation on the wire and enabling bi-directional streaming. Paired with the OpenConfig models, gNMI is SDN-ready.
- **gNOI** (gRPC Network Operations Interfaces): A collection of microservices that enable switch specific operations, like certificate management, device testing, software upgrade, and networking troubleshooting. gNOI provides a semantically rich API that replaces existing CLI-based approaches.

Trellis, as an SDN control application running on top of ONOS, controls packet forwarding across the switching fabric internal to a CORD POD (i.e., within a single site). But Trellis can also be extended across multiple sites deeper into the network using multiple stages of spines, as shown in Figure 6.3. This means Trellis has the potential to play a role in implementing the backhaul and midhaul network for the RAN, or alternatively, extending the RAN into customer premises (denoted "On Site" in the figure).

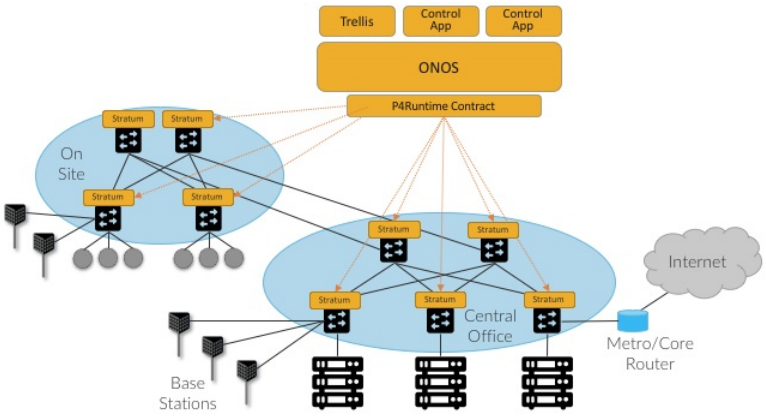

Figure 6.3: Trellis control application managing a (possibly distributed) leaf-spine fabric.

Chapter 7: Cloudification of Access

The previous chapters went step-by-step, first breaking 5G down into its elemental components and then showing how those components could be put back together using best practices in cloud design to build a fully functional, 3GPP-compliant 5G cellular network. In doing so, it is easy to lose sight of the big picture, which is that the cellular network is undergoing a dramatic transformation. That's the whole point of 5G. We conclude by making some observations about this big picture.

To understand the impact, it is helpful to first understand what's important about the cloud. The cloud has fundamentally changed the way we compute, and more importantly, the pace of innovation. It has done this through a combination of:

- **Disaggregation:** Breaking vertically integrated systems into independent components with open interfaces.
- **Virtualization:** Being able to run multiple independent copies of those components on a common hardware platform.
- **Commoditization:** Being able to elastically scale those virtual components across commodity hardware bricks as workload dictates.

There is an opportunity for the same to happen with the access network, or from another perspective, for the cloud to essentially expand so far as to subsume the access network.

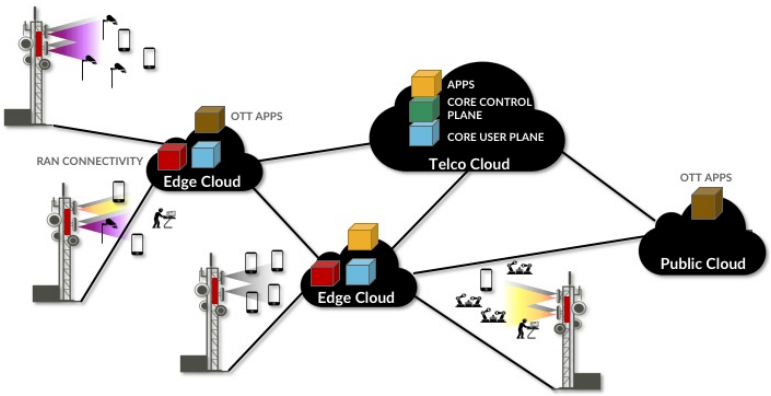

FIgure 7.1: A multi-tenant / multi-cloud—including virtualized RAN resources alongside conventional compute, storage, and network resources—hosting both Telco and Over-the-Top (OTT) services and applications.

Figure 7.1 gives a high-level overview of how the transformation might play out, with the global cloud spanning edge clouds, private Telco clouds, and the familiar public clouds. Each individual cloud site is potentially owned by a different organization (this includes the cell towers, as well), and as a consequence, each site will likely be multi-tenant in that it is able to host (and isolate) applications on behalf of many other people and organizations. Those applications, in turn, will include a combination of the RAN and Core services (as described throughout this book), Over-the-Top (OTT) applications commonly found today in public clouds (but now also distributed across edge clouds), and new Telco-managed applications (also distributed across centralized and edge locations).

Eventually, we can expect common APIs to emerge, lowering the barrier for anyone (not just today's network operators or cloud providers) to deploy applications across multiple sites by acquiring the storage, compute, networking, and connectivity resources they need.