AETOS: An Architecture For Offloading Core LTE Traffic Using Software Defined Networking Concepts

by

Kamraan Nasim

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the M.A.Sc. degree in
Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
Faculty of Engineering
University of Ottawa

Abstract

It goes without saying that cellular users of today have an insatiable appetite for bandwidth and data. Data-intensive applications, such as video on demand, online gaming and video conferencing, have gained prominence. This, coupled with recent innovations in the mobile network such as LTE/4G, poses a unique challenge to network operators in how to extract the most value from their deployments all the while reducing their Total Cost of Operations(TCO). To this end, a number of enhancements have been proposed to the "conventional" LTE mobile network. Most of these recognize the monolithic and non-elastic nature of the mobile backend and propose complimenting core functionality with concepts borrowed from Software Defined Networking (SDN). In this thesis we shall attempt to explore some existing options within the LTE standard to mitigate large traffic churns. We will then review some SDN-enabled alternatives, and attempt to derive a proof based critique on their merits and drawbacks.

Keywords-component; SDN; LTE; EPC; Openflow; mobile networks; deep packet inspection; network applications

Acknowledgements

I would like to acknowledge and extend my gratitude to my supervisor, Dr. Trevor J. Hall, for giving me the opportunity to work with him and for his continual intellectual support during my journey.

I would also like to thank my family, numerous as they might be, for their vote of confidence in me. In particular I would like to thank my mom and dad, Seema Nasim and Nasim Imtiaz, and my three lovely siblings, for their unrelenting faith in me. I will also like to extend a special thanks out to my grandmother Ms. Yasmin Habib, who took such a close interest in my thesis as if it were hers; and not to mention the nagging that came along with it.

I would also like to express my gratitude to Ms. Alessia Lerose, Mr. Vahid Hosseniounand Mr. Mohammad Aslan for their laughter, warm meals and peer reviews during my endeavor.

Table of Contents

List of Tables

List of Figures

Nomenclature

Abbreviations

AAA	Authentication, Authorization, Accounting
AETOS	Agnostic EPC Traffic Offloading through SDN
AIPN	All IP Network
ANDSF	Access Network Discovery and Selection Function
ARPU	Average Revenue Per User
CAPEX	Capital Expenditure
CDF	Cumulative Distribution Function
CN	Core Network
COTS	commercial off-the-shell
E-UTRAN	Evolved UMTS Terrestrial Radio Access Network
EnodeB	Evolved NodeB
EPC	Evolved Packet Core
EPS	Evolved Packet System

GBR	Guaranteed Bit Rate
GTP	GPRS Tunneling Protocol
HeNB	Home eNodeB
HSS	Home Subscriber Server
IMS	IP Multimedia Subsystem
IMSI	International Mobile Subscriber Identity
IRI	Intercept-Related Information
JFI	Jain Fairness Index
L-GW	Local Gateway
LENA	LTE-EPC Network simulAtor
LIPA	Local IP Access
LISP	Location-Identifier Separation Protocol
LTE	Long Term Evolution
MME	Mobility Management Entity
MNO	Mobile Network Operator
MTSO	Mobile Telephone Switching Office
NAT	Network Address Translation
NFV	Network Function Virtualization
NS3	Network Simulator 3
OAM	Operations, Administration, Management
ODL	Open Daylight

OF	OpenFlow
OPEX	Operational Expenditure
OSS	Operational Support Systems
OTT	Over-the-top
P-GW	Packet Gateway
PDN	Packet Data Network
QCI	QoS Class Identifier
QoS	Quality of Service
RAN	Radio Access Network
RNC	Radio Network Controller
RTC	Real Time Clock
RTT	Radio-Trip-Time
S-GW	Serving Gateway
SDN	Software Defined Networks
SDNMN	SDN Mobile Network
SFC	Service Function Chaining
SIPTO	Selected IP Traffic Offload
STP	Spanning Tree Protocol
TAI	Tracking Area Identifier
TEID	Tunneling End ID
UE	User Equipment

UML	Unified Modeling Language
VNF	Virtualized Network Function

Chapter 1

Introduction

We begin this work by outlining key trends that necessitate ongoing research in this area. In particular we focus on the problem that said research is attempting to solve. Having described the problem, we then argue the motivation to address it.

1.1 Problem Statement

With recent, significant strides being made in the realm of virtualization, cloud services and mobile computing, network planners and mobile network operators (MNOs) are being led to re-examine and reconsider traditional networking architectures. More than anything else, this re-focus is driven by the changing face of network applications. It is estimated that mobile gaming is growing at an annual rate of 19% grossing $13.9 billion worth of revenue annually [9]. Cisco stipulates that globally, mobile IP video traffic will hold a mammoth 79% share of all consumer Internet traffic by 2018, up from 66% in 2013 [23]. Innovation within the mobile network itself is considered as a key driver of this next generation of network applications. Compared to the previous 3G standard, LTE is:

- A faster radio access technology. Providing a peak downlink of 100Mb/s and Peak Uplink of 50Mb/s with speeds being even higher with LTE-Advanced.

- A flat, all IP based backhaul network (AIPN) which reduces total cost of ownership and provides less complex network and node deployments
- 3 to 5 times lower latency which facilitates a better user experience and enables more time-sensitive and mission critical applications
- A 2 to 3 times higher spectral efficiency

Another key facilitator is the global proliferation of data centers and cloud computing. The Cisco Global Cloud Index [39] stipulates that by 2019, more than 86% of workloads will be processed by cloud data centers. Global cloud IP traffic will more than quadruple over the next 5 years, reaching 719 exabytes per month. The cloud will empower mobile operators to elastically scale their networks based on the realtime demand instead of investing in network equipment that only addresses peak demands while remaining under-utilized during normal operations. However despite all of these facilitators, MNOs are stuck between a rock and a hard place: they must continue to meet expectations of the "vanilla" consumer who simply wants to use his or her device in order to make phone calls and send messages. At the same time MNOs will need to cater their networks to accommodate this new breed of data hungry consumers.

The original LTE network was not designed for this level of service differentiation. Yes, we have QoS and Evolved Packet Core (EPC) bearer configurations but they have more to do with resource allocation(such as in the case of Guaranteed Bit Rate (GBR) bearers) and prioritization on shared channels - the packet still traverses the same network. Given the centralized and non-elastic nature of nodes in the LTE core, it is not convenient for MNOs to reroute traffic on-demand to address network congestion issues.

1.2 Motivation

Conventionally, a reflex reaction to exorbitant demand is exorbitant supply. While this methodology has served MNOs well in the past, the mobile networks of today are at an inflextion point where average revenue per mobile user (ARPU) is not scaling in proportion

to network equipment capital expenditure (CAPEX). With consumers demanding faster, better and cheaper services, and MNOs competing with each other to win business, operators are averse to any change that requires a substantial investment within their network.

What is required is a way to radically re-think the problem. Existing research argues that Software-Defined Networks (SDN) may be able to overcome some of the drawbacks of LTE/EPC mobile networks by allowing network designers to have a logically centralized *controller* that can maintain a global view of the network topology and provide control elasticity. Openflow rules can be applied to redirect traffic to special nodes and thereby alleviate contention in different parts of the mobile network.

Our motivation for this work is to take two seemingly unrelated networking paradigms, and discuss mutually beneficial synergies. The overarching goal is to exploit techniques that shall seamlessly reroute traffic and mitigate traffic surges without compromising mobile operators ARPU metrics.

1.3 Contributions

The contributions made in this thesis are as follows:

- Proposing a framework for agnostic traffic offloading
- Implementing and Simulating a proof of concept design for non-intrusive traffic offloading and extracting key performance metrics to gauge the feasibility of this design.
- Deriving a relationship that uses load conditions to determine the favorability to trigger offloading.

It is our understanding that the evaluations done in this thesis demonstrate a comparable level of precision to similar works.

Chapter 2

Background and Related Work

In this chapter, detailed background information will be given on some of the components pertinent to this work. By doing so we shall set the stage for an in-depth discussion on Software Defined Mobile Network (SDMN) architectures. The first section of this chapter shall provide that overview. The second section of this chapter will build on top of where the first one leaves off, by presenting some of the related research from which our work draws intellectual inspiration. This section shall survey relevant papers in the field, and classify these works based on:

- the level of virtualization that they require; from a fully decomposed architecture to one that is only partially decomposed.
- the extent to which existing standard components will need to be modified.

2.1 Background

2.1.1 Long Term Evolution(LTE)

The LTE network can be divided into the radio front-end, referred to as Evolved UMTS Terrestrial Radio Access Network (E-UTRAN), and the IP backend referred to as Evolved Packet Core (EPC) [36]. The EUTRAN compromises of mobile devices, referred to as user

equipment (UE) which connect to base stations over the air interface. The base stations, or cells within the LTE standard are an evolution of what was commonly referred as NodeBs within UMTS and are therefore called Evolved NodeBs (eNodeB). The EPC, on the other hand, is responsible for providing control plane functionality and signaling; it is also a data anchor for connectivity to the Packet Data Network (PDN), and IP Multimedia Subsystem (IMS). In short, the EPC provides data and control services for LTE front-end components, namely the UEs and eNodeBs.

Figure 2.1 illustrates the core components of the LTE architecture and the interfaces between them.

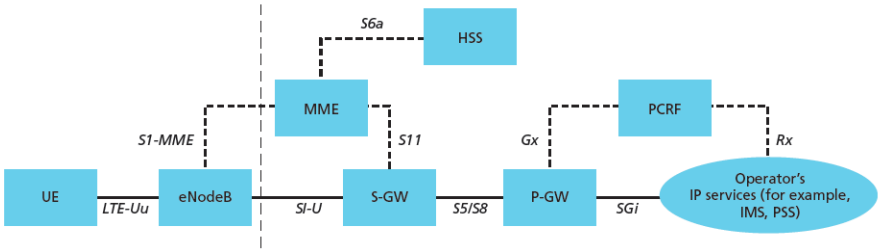

Figure 2.1: The LTE-EPC network architecture [36]

Our work focuses primarily on the EPC. Some components, relevant to this discussion, are:

- Mobility Management Entity (MME)
- Serving Gateway (S-GW)
- PDN Gateway (P-GW)
- Home Subscriber Server (HSS)
- GPRS Tunneling Protocol (GTP)

2.1.1.1 MME

The MME is essentially the "brain" of the LTE core network. Responsible for control and signaling functionality, some of its duties include UE registration, authentication and

bearer management(initial attach, paging). In addition the MME maintains UE context during mobility and inter-LTE handovers. The main functions supported by the MME can be classified as those related to management of EPC bearers, and those related to management of control connections or sessions.

To facilitate multi-vendor interoperability, the MME standardizes its interfaces. Table 2.1 outlines MME's interconnects.

Interface Endpoints	**Interface Name**
MME - EnodeB	S1-MME
MME - S-GW	S11
MME - HSS	S6a

Table 2.1: MME Interfaces

2.1.1.2 S-GW

The S-GW is the mobility anchor for inter-networking with other 3GPP technologies such as GPRS and UMTS. All UE IP packets to and from the UE are transferred through the S-GW. It is for this reason that S-GW is well suited to intercept user traffic for tariff and metering. The S-GW also buffers packets in downlink during a handover.

In addition to the S11 interface with the MME, the S-GW also provides a S1-U interface on which it receives user plane traffic from the enodeBs and redirects it to the P-GW via the S5/S8 interface.

2.1.1.3 P-GW

The second gateway, the P-GW, is responsible for allocating IP addresses for UEs and provides QoS and packet filtering facilities for flows that traverse through it. The P-GW is also the Mobility anchor for inter-networking with non-3GPP technologies such as CDMA and WIMAX. In addition to the S1/S8 interface with the S-GW, the P-GW also has an SGi interconnect with Operational Support Systems (OSS) such as IMS.

It must be noted that while S-GW and P-GW are logically very different, in most realizations they are usually kept on the same physical node, collectively known as S/P-GW.

2.1.1.4 HSS

The HSS contains subscription information for user accounts native to this core network i.e. the "home" users. This information includes, but is not limited to, the users QoS profiles, any roaming restrictions imposed by the MNO and which PDNs and MMEs this user can connect to.

The HSS can interface with an Authentication, Authorization, Accounting (AAA) server in the backend to provide user authentication services to the core network. The HSS can be considered to form the "edge" of the EPC core network and only has one south bound S6a interface to the MME, to which it provides subscriber and bearer information.

2.1.1.5 GTP

A 3GPP specific tunneling protocol called GPRS Tunneling Protocol (GTP) is used on all interfaces in the EPC core network. A remnant from the days of 2G circuit switching, GTP is used to multiplex sessions over a single tunnel such that it appears that each is a separate circuit.

GTP tunnels are established in a point-to-point fashion with the endpoints being identified by what is called a Tunneling End ID (TEID). What is meant by point-to-point is that a packet will traverse multiple tunnels for a single TCP or UDP connection. For e.g. when a UE data packet, being transmitted in Uplink, reaches the enodeB, it will first be encapsulated with the enodeB TEID as source endpoint and S-GW TEID as destination endpoint. On reaching the S-GW, the GTP header will be decapsulated and a new one added, with P-GW as destination endpoint, for the remaining leg of the trip.

Figure 2.2 illustrates the point-to-point nature of the GTP tunnels. This embodiment

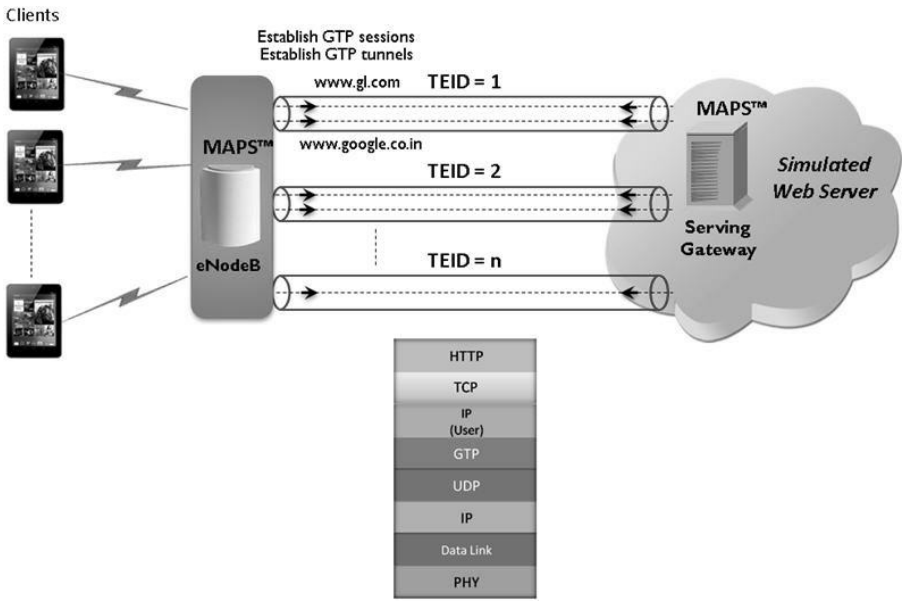

Figure 2.2: enodeB - S-GW point-to-point GTP tunnels and protocol stack [10]

also depicts the OSI stack of a GTP packet. As can be observed, the GTP header is encapsulated on top of layer 4 UDP (or TCP) header.

Suffice to say GTPs are central to the flow of information through an LTE core network and will play a major role in the forthcoming sections within this thesis.

2.1.2 Network Function Virtualization

Network Function Virtualization, or NFV as it is more commonly known, has been heralded as the next big thing in the networking world. Simply put, NFV is the principle of moving host based network functions and applications from dedicated hardware onto virtual instances running on commercial off-the-shelf (COTS) server either on-premise or in the cloud [21]. Traditionally, a node within a mobile network refers to a dedicated hardware box that is single-service, and with a very well defined functional behavior, with proprietary inner workings albeit standardized external interface [44]. NFV is a clear deviation from this mind-set.

The key driving force behind NFV, is its ability to scale capacity on-demand. Setting up

a new network node is as simple as spinning a new virtual instance without going through lengthy qualification cycles or steep learning curves as is characteristic of new hardware deployments. MNOs have wholly embraced this concept since it allows them to scale their infrastructure during peak demands or certain seasonal trends, without permanently investing in hardware and rack space on-premise.

Another major leg-up in NFV's adoption is vendor neutrality - by moving away from proprietary network hardware, MNOs are no longer locked in with a given vendor and can build a truly multi-eclectic network one that maximizes their shareholder value without compromising their customer experience.

Given that most data centers hosting cloud networks are multi-tenant; infrastructure and administrative staff and utility costs are split and shared, thereby reducing operational expenditures(OPEX) for each of these MNOs had they hosted these network functions on-premise.

Figure 2.3: **LTE network topology depicting both conventional EPC and an NFV based EPC**

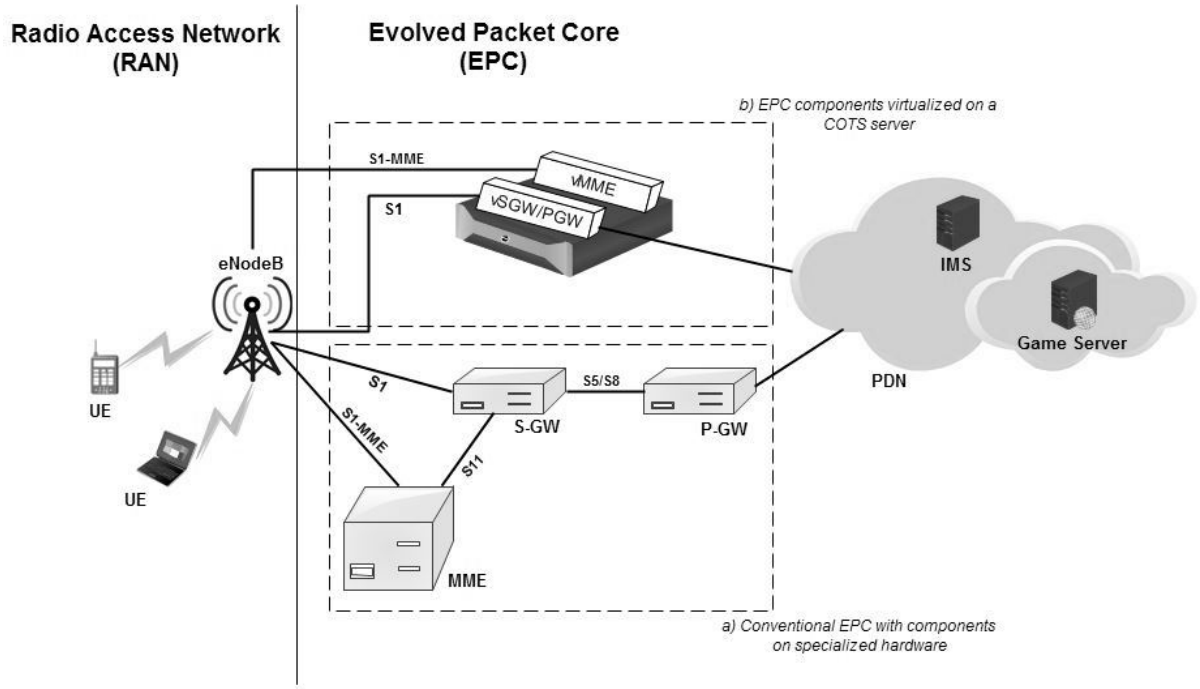

Figure 2.3 is our depiction of a vanilla LTE-EPC architecture, with core components deployed over dedicated hardware, and how it would look like if said components were virtualized on a commercial off-the-shelf (COTS) server.

2.1.3 Openflow

Openflow (OF) is the first standardized southbound communication interface between the control and the forwarding layers of an SDN. Openflow allows a logically centralized controller unabated access to the the forwarding table of network devices(such as switches and routers in the classical sense of the term). The protocol specifies commands to *add*, *modify* or *delete* flow rules within the flow table on the network device [8].

Each installed flow rule contains a *Match* portion that dictates what to look for within incoming packets, and an *Action* that tells what to do incase there is a match. A matching condition must be some attribute of the incoming flow that identifies it. In the case of IP packets, these are the five tuple parameters(source IP, destination IP, source Port, destination Port, transport protocol). In the case of L2 packets, these conditions may be the VLAN ID, source MAC or destination MAC. Each parameter of the packet header may explicitly specify a matching condition(such as 'srcIP = X.X.X.X'), or may use wildcards to facilitate aggregation of flows. For e.g. a flow rule with 'srcIP = *' is essentially a *Don't Care* for the source IP address of a packet and will match all packets as long as other matching conditions are satisfied.

Table 2.2 represents fields from packet headers that constitute our matching conditions.

Ingress Port	meta data	src MAC	dst MAC	VLAN id	VLAN pri-ority	MPLS tag	src IP	dst IP	IP ToS	L4 src port	L4 dst port

Table 2.2: Openflow rule matching conditions

As each packet traverses an OF enabled device, it is sequentially compared against each flow rule in the flow table for a possible match. As is quite likely, a packet may match multiple rules within the table, in which case the rule with the highest priority will take

precedence and will be executed. In case there is no match, the default rule in the tale is the one that is executed on the packet. In that sense, the flow table lookup has a worst case O(n) delay. Once there is a match, the standard allows for a number of possible actions [37] [24]:

- Forward this matching packet to one or multiple output ports
- Encapsulate and forward this matching packet to the controller. As an optimization only the packet header may be forwarded to the controller, if the latter requires the entire packet then it may request so using the packet buffer Id
- Drop this matching packet
- Forward this matching packet using the switch's normal forwarding table
- Forward this matching packet to a group defined in a group table which will then decide what further actions need to be taken on this group
- Forward this matching packet to another rule table. This principle is important as it can allow network designers to develop pipeline models for spreading rules across tables and optimize the O(n) delay for any individual flow rule table.

It must be noted the actions outlined above represent the bare minimum requirements for a switch to be classified as Openflow enabled. Most OF switches these days, support additional flow actions such as packet header re-write operations, which enables Network Address Translation (NAT) deployment scenarios, pushing/popping MPLS tags, and QoS class demarkations(such as CoS or ToS/DSCP markings[1])

To put things in perspective, the aforementioned *rule* in the rule table comprises of: (1) a header indicated the packet matching conditions, (2) the action to undertake, and

[1] Class of Service(CoS) applies to Layer 2 traffic. Similarly Type of Service(ToS) and its successor, the Differentiated Service Code Point(DSCP) apply to IP packets. The purpose for all of these is similar: to mark the packet as belonging to one or more well defined traffic classes, in order to provide differentiated QoS. For more information on this, refer to http://www.rhyshaden.com/qos.htm

(3) statistics counters indicating the number of packets that have been matched as well as timestamps indicating when was the last match and how long this rule has been active.

The Openflow standard supports a number of message primitives. These can be broadly categorized as *controller-to-switch*, *asynchronous* and *symmetric* messages. As the name indicates, controller-to-switch messages originate at the controller and can be either queries or commands. Conversely, asynchronous messages are initiated by the switch. Symmetric messaging represents the middle ground which either controller or switch can initiate without prior solicitation[24]. Table 2.3 will attempt to outline some of the subtypes within these categories. Numerous as they are, in the interest of brevity we will confine ourselves to only those relevant to this thesis.

Message SubType	*Description*	*Category*
Feature	requesting the capabilities of the OF switch	**controller-switch**
Modify-State	flow or group table or state modification	**controller-switch**
Read-State	stats query messages	**controller-switch**
Packet-Out	direct controller originating packets out a switch port or forward switch Packet-In packets after processing them	**controller-switch**
Packet-In	used by the switch to send a packet up to the controller	**asynchronous**
Port-status	switch port state messages sent up to the controller	**asynchronous**
Error	standard error reporting up to the controller	**asynchronous**
Echo	keep-alive messages that can be sent by either a switch or a controller to check if link and receiver are still active	**symmetric**

Table 2.3: OF message subtypes and their descriptions, organized by category

2.1.4 Jain's Fairness Index

A fairness index is a quantitive measure in a distributed system that determines if allocation of a shared resource is being done fairly and if not then what percentage of users are being affected by this discriminate allocation. On the subject of Network Engineering, a number of such measures are available in academia of which Jain's Fairness Index (JFI) [30] is the most prevalent. JFI offers a number of salient advantages:

- it is scale and metric independent unlike other fairness methods such as Variance and Standard Deviation.
- it is bounded between 0 and 1, with 1 implying fairness for all and 0 implying fairness for none.
- it is a continuous measure, where an update in the input candidate list will result in a change in the fairness index
- it is broad and generic and can be applied to any shared resource.

For a set of n users such that the resource x is shared between them and x_i is the resource allocation for the i^{th} user, JFI can be written as:

$$f(x) = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} (x_i)^2} \left\{ x_i =\geq 0 \right. \tag{2.1}$$

Another way to look at JFI is to compute a *fair allocation mark* i.e. the amount of allocation x_i for the i^{th} to feel that it has been treated fairly.

The *fair allocation mark* can be computed as:

$$x_f = \frac{\sum_{i=1}^{n} (x_i)^2}{\sum_{i=1}^{n} x_i} \tag{2.2}$$

The allocation of each user is compared against the fairness mark and deemed as *fair* or *discriminate* depending on if it is higher than the fairness mark or lower. The combined JFI is the average of these individual comparisons:

$$f(x) = \frac{1}{n} \sum_{i=1}^{n} x_i / x_f \tag{2.3}$$

Given the relative freedom in the choice of the shared resource, that JFI affords, we shall employ this index for our analysis in this thesis.

2.2 Related Work

Before delving into recent initiatives in traffic offloading relevant to SDNMNs, it is important that we have a look at existing offloading and redirection techniques within the LTE standard.

We would like to remind the reader that the scope of this thesis restricts itself to such techniques within the EPC core network only. Traffic offloading within the radio network is a broad topic that merits its own independent study. Suffice to say, and without embarking into too much detail, Wifi based offloading is the de facto technique for congestion alleviation with the radio access network. To this end, Access Network Discovery and Selection Function (ANDSF) was defined in recent 3GPP standards as a framework for issuing policies to a device where traffic routing decisions between a cellular and WiFi access network are being made [11].

2.2.1 Existing Offloading Techniques in EPC

The concept of traffic offloading has its proven benefits and is well noted within the LTE standard. It is estimated that by 2017, more than 30% of core network traffic may be offloaded to fixed IP networks [27]. Indeed both 3GPP Release 10 and Release 11 specifications have proposed various offloading techniques, to address contention in both EUTRAN and the EPC core network.

For traffic offloading within the EPC core network (CN), the following are some standardized techniques available to MNOs [42]:

2.2.1.1 Local IP Access(LIPA)

LIPA based offloading is only usable on LTE femtocells, or Home EnodeBs (HeNBs) as they are also known. These are limited bandwidth, low power transmitters meant for home or campus deployments. A local gateway (L-GW) , is connected to the HeNB and allows UEs, attached to this HeNB, the ability to bypass the core network and access certain services over the local fixed IP network. L-GW must support limited PGW as well as SGW functionalities such as UE IP address allocation(for the local PDN), Downlink Packet Buffering, Paging etc. The L-GW connects to the EPC S-GW via a L5-S5 tunnel interface. L-GW Control messages intended for the MME are relayed through this S-GW.

LIPA makes sense if and when the MNO provides its subscribers certain geo-location based (or intranet) services, for which traversing the CN is inefficient. LIPA posed an interesting business case for Over-the-top (OTT) players such as advertisers and content providers to offer specialized value-added content to subscribers using their femtocells.

Promising as it might sound, LIPA does not come short of its limitations. For one thing the facility is not available on regular enodeBs i.e. macro cells. Also, while 3GPP Release 12 has added provisions for mobility, service continuity is still fairly limited, i.e. every time the UE moves away from the femto cell, all sessions are terminated. Even more so is the fact that LIPA provides user congestion alleviation at the expense of increased signaling in the EPC core network. The initial UE message to the MME must now includes a information element to indicate the IP address of the L-GW. Also all paging requests from the L-GW to the S-GW are forwarded to the MME and must be serviced there. This additional signaling must be done for every L-GW, which might be as numerous at the number of HeNBs in the MNO's RAN, in the event that L-GWs are co-located within the femtocells.

We posit that this is one area that would benefit from a logically centralized SDN control. Presence of such a controller shall take away compute complexity from the HeNBs in setting up the the L-S5 tunnels and L-GW IP Addresses and shall do so globally for a cluster of geographically close femtocells.

2.2.1.2 Selected IP Traffic Offload(SIPTO)

Unlike LIPA, SIPTO can be used on both LTE macrocells and femtocells. SIPTO involves the MME selecting a (S-GW, P-GW) pair geographically close to the UE's point of attachment in the RAN thereby minimizing the UE's round-trip-time (RTT).

Subscription information on SIPTO eligible UEs is stored in the MNO's home subscriber server(HSS). We would like to point out that here lies a key distinction between LIPA and SIPTO. In the case of the former, the decision to offload is made by the HeNB which may or may not be owned by the same operator as the core network. Whereas for the latter, this decision is centralized at the MME. SIPTO leverages this centralized selection to provide service continuity to its UE subscribers during mobility and handover. However, similar to LIPA, relief in the user plane comes at the cost of increased signaling within the core network. In addition to the initial gateway selection in SIPTO, the MME needs to monitor UE tracking area updates and initiate explicit detach followed by a reattach procedure for gateways that end up being closer to the UE.

It is our perception that SIPTO is one area which would also benefit from existing SDNMN research and solutions that propose moving MME functions within a logically centralized control. The additional signaling involved in SIPTO is further exacerbated by increased east-west communication between MME nodes when a UE context(SIPTO subscribed) is transferred from one MME to another. Have a global state machine to issue SIPTO policies and make decisions would cut through some of these east-west exchanges.

2.2.1.3 S1-flex

S1-flex is an LTE standard concept that allows MNOs to create a pool of MMEs. Individual eNodeBs can then have multiple concurrent S1 connections to the MMEs within the pool as opposed to a single S1-MME link as is the conventional case. Although S1-flex is not a "traffic" offloading technique per se, and some would even argue that it is akin to MME failover redundancy, it does provide operators the ability to offload EPC signaling traffic to a bypass core network which offers up its MME within the common MME pool. It is

also for this reason that S1-flex is a popular deployment option for multi-operator RAN sharing where MNOs contribute to an MME pool and have RAN macro cells multi-homed onto this pool.

The technology offers a number of merits to MNOs. Sudden imbalances in control traffic or over commitment of an MME node can be addressed without detaching the enodeB or loosing UE context. S1-flex allows for seamless migration and load equalization with the MME pools. Secondly since UE context is shared between the MME nodes of a pool, it reduces the number of Tracking Area Update exchanges between the MME and HSS. If a UE moves within its pool area, call processing can still be done without registering the new location update with the HSS.

It is worth mentioning that while LIPA and SIPTO focused on user traffic offloading, S1-flex is a purely control plane offering. However the mechanism does have some perceived shortcomings. For one, the flat distributed architecture of LTE would imply a M:N mesh of S1-MME connections. The guarantee of failover protection and control redundancy comes at the expense of significantly more chatter on the wire and within the operator's backhaul network. Having a centralized controller, such as the Radio Network Controller (RNC) from the days of UMTS, to police S1-flex negotiations would reduce complexity at the enodeB and cut down on some of this chatter and allow for tighter interaction and migration between different MME pools.

Table 2.4 summarizes some of the characteristics of these three offloading techniques. All mechanisms discussed above have their merits and are prevalent in industry. However they are not mutually exclusive. MNOs may opt to have all three schemes present in their networks.

Figure 2.4 embodies this approach. In this illustration, UE1 connects to a HenB which provides LIPA based offloading via the L-GW. This L-GW can allow connectivity to a fixed IP network(not shown in the figure), that may provide intranet based services such as video content distribution and news groups(depending on the business case), as well

Characteristics	**LIPA**	**SIPTO**	**S1 Flex**
RAN limitations	not applicable for LTE macrocells	no limitations	no limitations
UE mobility support	no mobility support	supports mobility	supports mobility
Traffic offloading traffic	Home EnodeB	MME	MME/EnodeB
Type of traffic offloaded	user-plane	user-plane	control-plane
Perceived Benefits through SDN	logically centralizing L-S5 tunnel creation and L-GW IP address allocation will reduce processing complexity at the HeNB and promote mobility	improves MME's S-GW/P-GW selection function and improve UE context transfers across MMEs	reduces the number of concurrent S1-MME connections and reduces process complexity at the enodeB

Table 2.4: LIPA vs. SIPTO vs. S1 flex comparison table

as provide the core network via S/P-GWa. UE2 connects to the enodeB which should normally have interfaced with S/P-GWa for user plane connectivity; however since UE2 is SIPTO subscribed, the MME recognizes that S/P-GWb is geographically closer to it and selects that gateway pair instead. Finally, in this embodiment, notice that the enodeB connects to multiple CN sites. This multihoming reflects the benefits of S1-flex where control plane traffic may be load balanced across redundant S1-MME connections.

The next section however will survey some SDNMN based offloading solutions that truly challenge the status quo.

2.2.2 EPC Offloading Techniques that leverage SDN concepts

In general, traffic offloading in the LTE network is not a novel concept. However, as is usually the case with performance optimizations, offloading in the EPC core network is an after-thought and suffers from some of the same ailments as the LTE network. For one,

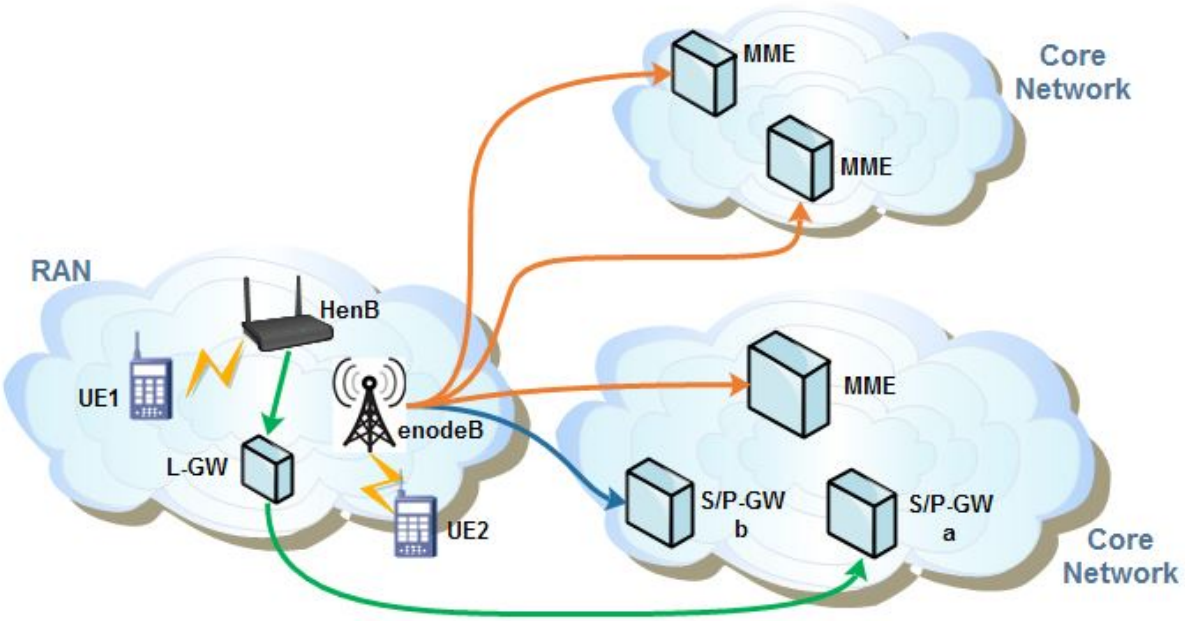

Figure 2.4: Deployment depicting LIPA, SIPTO and S1-flex traffic offloading approaches. The arrows in Green represent LIPA where-in HenB interfaces with L-GW. The workflow in Blue represent the fact that UE2 is SIPTO subscribed and while normally its enodeB should have connected to S/P-GWa, in this case the MME selects the closer S/P-GWb. Finally the flow in Orange represent S1-Flex where the enodeB connects to an MME pool.

performance gains in user plane come at the expense of an increase in signaling complexity. Owing to the centralized nature of gateway nodes within the core network, and usually at the network edge, UE flows have to traverse large distances to reach web servers that may be even be located geographically close to them. Having a controller with a global network view can improve this suboptimal routing by improving coordination between MME nodes. In addition, with NFV technology, virtualized gateway nodes can be spawned gratuitously nearer to the RAN.

The following sub sections will review some state-of-the-art works within the confines of EPC based offloading and highlight some common themes amongst these works. These sections have been laid out in a progressive manner, where one theme transitions into the next.

2.2.2.1 Fully Decomposed vs. Partially Decomposed NFV

We begin our SDNMN survey by mentioning the study done by Basta et al in [15] which is one of the earlier works in this area. The authors propose four possible deployment scenarios for full or partial decomposition of S-GW/P-GW functionality as virtualized network functions(VNFs) in the cloud. These designs propose replacing all switches and forwarding elements within the Operator's backhaul network with Openflow enabled switches that use a logically centralized SDN controller to steer the traffic from the core EPC network to the cloud network. Functions such as policy control, packet filtering and charging are kept within the Openflow switches. The authors propose using Openflow flow counters and statistics messages, as defined in the specification, to realize offline and online tariff charging.

Figure 2.5 illustrates Basta's decomposition scenarios which depicts functional placement of EPC functions either in a datacenter/cloud or co-located in the user-plane with the OF switch. All of Basta's designs presume that the mobile backhaul, connecting the EPC components, is made up of Openflow enabled switches. In that respect, an SDN controller is expected to install OF rules that steer mobile traffic from the core network to the cloud site.

Figure 2.5a is a full cloud based migration scenario where all EPC components are virtualized and moved to an on-premise or data center based cloud environment. This option brings with it all the benefits of the cloud, namely elasticity, scalability and fault tolerance at the expense of performance penalty since data plane traffic is no longer being forwarded at line-rate. Figure 2.5b recognizes this limitation and proposed only decomposing the EPC control plane, namely the entire MME and HSS functionality, and the partial S-GW signaling features in the cloud. All data plane components such as P-GW, Policy Control and Rate Charging Function (PCRF) are confined to the legacy core network which inturn satisfies the stringent performance requirements of data traffic. However this design may cause a bottleneck in the cloud, and a single point of failure, as the entire control plane has been decomposed there. Figure 2.5c is a variation on (b) in which only the MME is decomposed in the cloud and all S/P-GW functionality is kept as is, i.e. in the core network.

Finally the embodiment in Figure 2.5d tries to achieve the best of both worlds by cloning the entire EPC in the cloud, with the RAN multi-homed onto both the legacy CN as well as this cloud based EPC. The beauty of this solution lies in the fact that delay tolerant sessions, with less stringent QoS requirements can be offloaded into the cloud based EPC and benefit from the greater compute and storage capabilities that the cloud provides; while delay sensitive traffic will still be handled in the legacy manner by the core network. The complexity of this solution lies in the state synchronization required between the cloud based EPC and the core network and which the SDN controller will need to shepard.

It is our opinion that a fully decomposed S-GW/P-GW design may not be feasible owing to stringent latency and bandwidth requirements in the data plane. Since both EPC control and data plane traffic is encapsulated as GTP packets, either the forwarding OF switches must parse these GTP headers or these switches must send those packets north-stream to the controller which in turn has GTP parsing logic built into it. The latest Openflow standard as specified in [43] has added support to identify tunneling packets, and allow for the use of tunnel metadata associated with those packets as Matching conditions; however

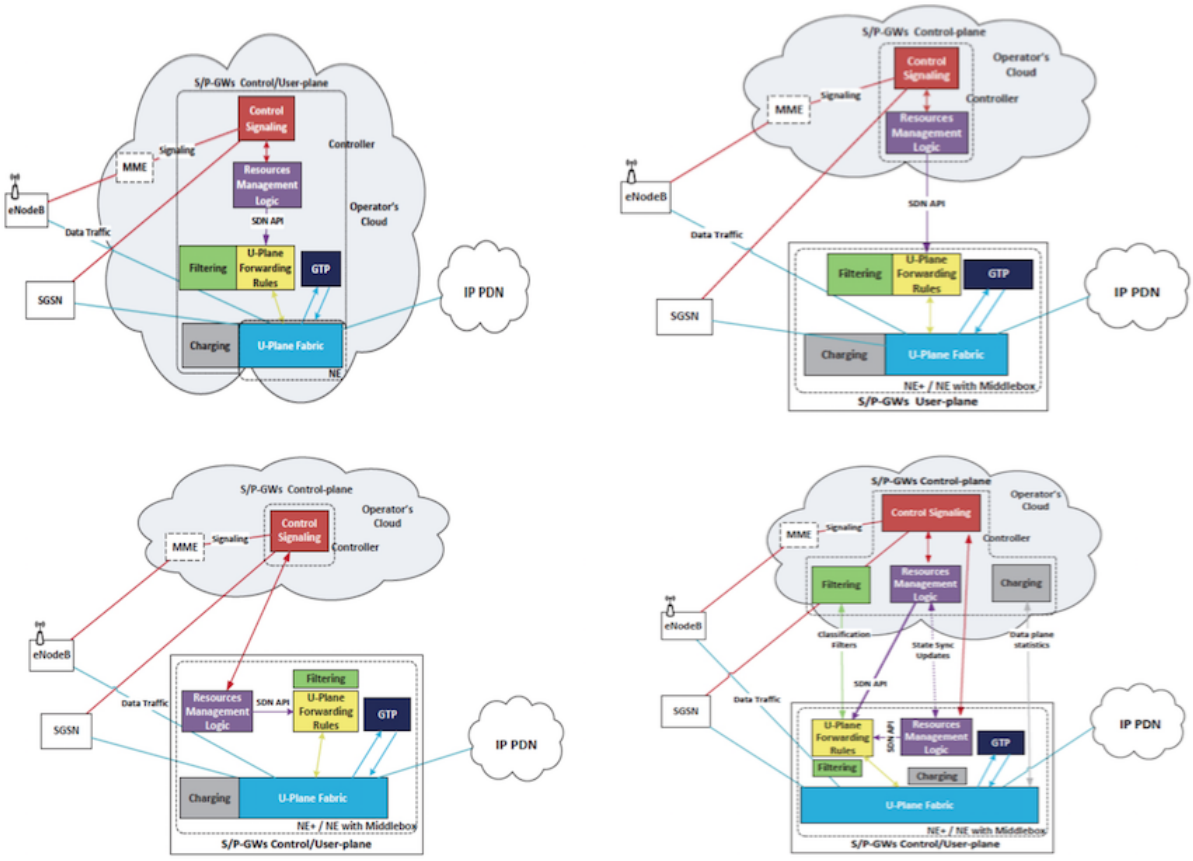

Figure 2.5: Architecture reference models of four decomposition options envisioned by Basta [15] [L-R in clockwise] a) EPC fully decomposed in the cloud, b) EPC control plane fully decomposed in the cloud, c) EPC control plane partially decomposed in the cloud, d) EPC clone fully deployed in the cloud

this provision is only limited to VxLAN, MPLS and GRE tunnels for now. We therefore know for a fact that all such GTP matching logic needs to be baked into the controller.

We argue that doing so will incur a substantial performance penalty for the data plane as this GTP processing is not being done at line rate within the forwarding element and instead sent over the wire to the Openflow controller which now becomes the performance bottleneck. In contrast, control plane traffic is fewer and further in between and has more relaxed timing requirements. GTP parsing for such traffic therefore seems like a more feasible option. In our opinion the hybrid design in 2.5d seems to be the most pragmatic approach and resonates with some of the principles outlined in section 2.2.2.3.

2.2.2.2 Partially Decomposed S-GW

Nevertheless, we cannot discount the fact that Basta's study paved the way for a number of interesting follow-on proposals in this area. One of which was [16] in which Said et al focus on resiliency, and load balancing within the Mobile network.

With an overall view to promote on-demand connectivity within the network, the authors recognized the benefits of partially decomposing EPC functionality within the cloud and propose an architecture where the entire intelligence within the SGW (SGW-C) and MME is centralized as applications running on top of an Openflow controller. Figure 2.6 outlines their proposition. To this end, the authors replace the GTP based control protocols that run on the S1-MME (between MME and eNodeB) and S11 (between MME and SGW) interfaces by the OF protocol. All aspects of SGW responsible for user plane and data forwarding (SGW-D) are still kept within the MNO's core network, as well as all other EPC components such as PGW and HSS. Therefore all user plane protocols stay the same. The SGW control plane is responsible for GTP tunnel establishment and TEID allocations for user sessions. The key benefit of consolidating SGW-C functionality above the SDN controller, is that a user session may be load balanced between SGW-D sessions without having to reallocate a TEID for the session and initiate S1/S5 Bearer Release signaling procedures. The TEID value is therefore allocated once per session within the SGW-C and remains invariant during UE session mobility across SGW-Ds.

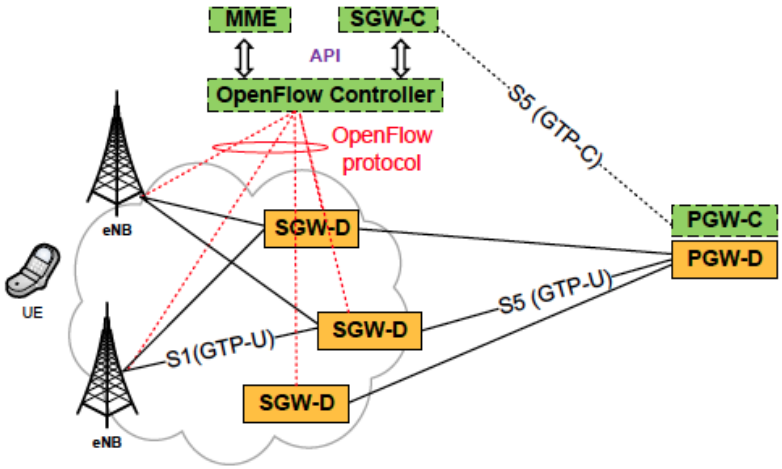

Figure 2.6: OF based EPC architecture with S-GW partially decomposed [16]

In a similar fashion(to Basta's work), Said et al leverage the OF counters and statistics messages as a means for the SGW-C to monitor its SGW-D array and make dynamic load balancing and offloading decisions. The authors go one step further by contrasting a SGW failure scenario in a conventional LTE network and then with their modified architecture. The latter clearly depicts a significant saving in signaling because new GTP tunnels do not need to be established.

In some aspects, we feel that Said's design can borrow concepts from SIPTO and make the decision to offload a UE session to a new SGW-D based also on the UE's geo-proximity to a SGW-D (and not just load statistics)

2.2.2.3 UE Agnostic Traffic Redirections

All things considered, the concept of full decomposition of an EPC component in the cloud is still an interesting viewpoint, and one which is exploited by Banerjee et al in their MOCA architecture [14].

In MOCA, UE uplink data is offloaded to an on-premise cloud network that contains virtualized SGW and PGW nodes that receive UE traffic and transmit the corresponding Downlink traffic. This offload is meant for specific UE sessions only. The authors propose online gaming as being the business case that will drive adoption of this offloading strategy;

although the architecture is general enough to be applicable to a number of applications. Unlike the previous work in [16] (Said et al), MOCA does not swap out LTE's native control protocols with Openflow but does require a number of changes to the MME in order to drive dynamic offloading of specific mobile traffic.

In order to determine the UE session that needs to be offloaded, and the enodeB - SGW endpoints for this session, the MME will need to extract UE IP address, the SGW TEID, the enodeB IP address and the enodeB TEID from the message primitives involved in a UE attach signaling procedure. The MME will need to maintain a mapping between the UE IMEI and these extracted parameters, for use when offloading is required.

It must be noted here that inspection of EPC control traffic to extract these offloading primitives is not a requirement uniquely imposed by MOCA. Government legislations require MNOs to provide network traffic information to law enforcement agencies. Lawful interception is therefore a main feature of most EPC networks. Amongst other types of information, MNOs must provide Intercept-Related Information (IRI) for UE attachment and detachment procedures [34] which is what MOCA needs to extract the offloading attributes.

Although not indicated by the authors, it is our understanding that these mappings will be stored in the HSS along with other subscriber information. Localized storage at MME is not recommended since during MME transfer such as S1-flex procedures, these mappings will need to be moved to the new MME which is an additional overhead. Offloading is initiated via an external trigger to the MME, from an offloading front-end which in the author's view is the game server front-end. The trigger indicates the UE IMEI that needs to be offloaded. The MME will therefore need to be modified to handle these message primitives as well as opening an upstream interface to the front-end. If the offloading front-end is an MNO provided service then we may contend with using one of the existing interfaces such as the S6a interface with the HSS, however in the case that this is a 3rd party service, such as the gaming server use-case that MOCA's authors have stipulated, then we do have some interoperability and security concerns, since it is our impression that MNOs would not be too keen to open up a critical control component to external service

providers and OTT players.

The authors of MOCA realize that Openflow switches in their current stage do not support GTP header parsing, and therefore propose using middleboxes that parse the GTP header for packets in Uplink and redirect the flow to the offload cloud network if it belongs to a UE that has been marked for offloading. Similarly, the middlebox will receive downlink packets from the cloud based S-GW but must rewrite their outer source IP address to that of the original S-GW. This is because unlike some of the previously discussed techniques which require explicit signaling to register the new S-GW, both the UE and enodeB are agnostic to the offloading and assume the packet comes from the registered S-GW.

Finally, although not explicitly indicated by the authors, it is our understanding that MOCA is meant for UDP based sessions only, as TCP requires explicit handshaking between the UE and the cloud based PDN which will not work as the flow redirection is transparent to the UE. Given that the paper targets only online gaming applications, we can assume that the authors implicitly assume only UDP in their design which is a noteworthy limitation in our opinion.

After a close examination of MOCA's design, one may question the need for an offload S-GW and P-GW, if the intended use-case is simply to connect to a local PDN. Banarjee et al indicate clearly that the new S-GW will not respond to paging requests as its presence is oblivious to the UE; nor can we expect it to be a mobility anchor for inter-3GPP communication. The new P-GW is also a fairly thinned out version of the original thing - it will not be allocating UE IP addresses and likely not providing inter-connect with non-3GPP technologies. The offloading network is simply a fixed IP network, very similar in kind to the local network provided to UEs in LIPA. In the usability context that MOCA has stipulated i.e. online gaming and other delay-sensitive applications, we feel that providing fully decomposed gateways is a bit of an overkill. And sure enough, our perspective resonated with the authors of SMORE [22].

Cho et al, whose work is based on MOCA, recognized that any modifications to the standard MME including opening it up for 3rd party interaction, is a path of maximum

MNO resistance and thereby proposed an alternative design that is identical to MOCA with the exception of the offload network not containing full-fledged S-GW and P-GW nodes. Accordingly, the MME does not need to be modified to send overloaded Create Session Request messages. Also the external offload trigger now comes to a separate component within SMORE's architecture called the SMORE controller which for all intents and purposes may be an application running on top of an SDN controller. SMORE seems to be a more pragmatic approach to the issue of UE agnostic traffic offloading.

One common theme in both MOCA and SMORE is that offloading is caused by an external input. As we mentioned previously, there are certain interoperability challenges in providing non-standard interfaces within EPCs. Given the high value nature of networks today and sensitive information they carry, operators are apprehensive about any and all changes not dictated by the LTE standard. Most MNOs would argue that the trigger to offload must come from within the network itself.

The authors of ProCel had a similar idea in [38]. Nagaraj et al claim that upto 70% of traffic in today's networks is delay tolerant and is either data or video-demand-traffic, which does not require stringent QoS guarantees. They argue that said traffic does not require seamless L3 mobility since it is loss tolerant and neither does it normally require inter-networking with other 3GPP technologies. Traversing such traffic across the core EPC network overburdens the network mainly because of the large signaling overheads associated with setting up GTP tunnels for these sessions that may not necessarily benefit from it. To this end, ProCel attempts to proactively classify all flows in the core network into either core flows, that will continue through the EPC as usual, and non-core flows that will be offloaded to a fixed IP network. The authors indicate in their work that signaling is growing 50% faster than data traffic within LTE networks and adopting such bypass techniques will help to alleviate some of these loads.

We would however like to point out that all three designs, namely MOCA, SMORE and ProCel, suffer from a major pitfall. Concerning GTP parsing, these designers recognize the absence of inline matching on the OF switch and propose using middle boxes as a remedial measure. However the exact ramifications of doing so are not discussed. As we mentioned

previously, the lowest granularity afforded to us by an Openflow rule is the IP five tuple. For UE traffic this becomes:

- in the case of Downlink: source IP address is the S-GW IP and destination IP address is the enodeB IP.
- in the case of Uplink: source IP address is the enodeB IP and source IP address is the S-GW IP address

Since OF rules do not support bidirectionally at this time, two rules will need to be installed for both the downlink and uplink directions but with different flow actions. What is implied here is that the flow will match all packets between the same endpoints and therefore be steered towards the middlebox which will then decapsulate the GTP header and inspect the inner header to determine if this packet is indeed for a UE session that was marked for offloading. If it turns out that this was indeed the case then the effort was not in vain and the packet can be forwarded to the offload network. However, as will be the general case, most packets will belong to sessions that should not be offloaded. This introduces an additional latency which may be highly undesirable for delay-sensitive sessions such as GBR bearers used for conversational voice, VoIP and video conferencing. What is worse is the fact that the middlebox will have to send all these non-matching packets back to the ingress switch so that they can be forwarded normally to their original destinations. The switch will need to turn off Spanning Tree Protocol (STP) since the return interface from the middlebox back to the OF switch will be detected by STP as a bridge loop which will then shut off that port causing all of those packets to be dropped. The other option is for the OF switch to copy the packet before steering it to the middlebox, the latter of which can then safely discard it if it were not meant to be offloaded. However packet copy operations in the fast path are quite expensive performance-wise. In the context of the Openflow[24], it may make more sense to delegate the middlebox as an EQUAL or SLAVE controller, connected to the switch not through its switch port but through the Control Channel. Another option described in the standard is Port Recirculation which can be leveraged while keeping the middlebox connected to a switch port.

In our understanding, building GTP parsing logic inside an Openflow controller entity that is co-located with the OF-switch may be a worthy compromise since the OF switch does not need to send the entire packet to the controller but only the header and a buffer Id, and these too shall traverse a shorter distance. The OF controller can offset some of the latency introduced for non-offloading flows by rewriting the output port to that of the destination or the next hop switch, thereby saving some cycles had the non matching packet been processed through the switches L2 forwarding table. The implications of this realization are severe: *if an enodeB has N UEs attached to it and only 1 of them is marked for offloading then all remaining N-1 UEs will incur a performance hit.* Grounded in this reality, we do believe that the problem is slightly less worse for ProCel when compared to MOCA and SMORE. Since ProCel impartially tries to classify all flows, and not just those traversing a certain enodeB, the additional delay introduced is a network constant and is deterministic, allowing MNOs to factor it apriori.

2.2.2.4 Middleboxes

While we are on the subject of middleboxes we need to mention SoftCell as outlined in [31]. The authors draw reference to some of the offloading techniques that we have mentioned, and subsequently argue that any form of OF rules installed in the backhaul to dynamically steer traffic is not a scalable work model owing simply to the large number of enodeBs, and an even larger number of UEs in a typical access network.

What is needed, is some way to classify flows under a higher order hierarchy, and have a common action that applies. This will allow switch forwarding tables to scale much better. In a concept somewhat similar to BGP prefix aggregation, SoftCell aggregates flow rules across three dimensions, a) service policy, b) network location, and c) UE id. Service policy aggregation would allow MNOs to create a policy tag and aggregate multiple flows in it. In location based aggregation, the SoftCell authors borrow principles from Location-Identifier Separation (LISP) protocol [28] and UE IP addresses are mapped to location identifiers which serves as another means to assimilate UE flows. Finally, SoftCell claims that classifying flows by UE IMSIs provides positive value for MNOs since most intrusion

detection and other Operations support systems (OSS) would apply common policies to all traffic from a particular UE or in some cases, block traffic from a particular UE all together.

This begs the question as to how MNOs can build these higher order aggregations when the only building blocks they have are the IP five tuple attributes or L2 attributes. SoftCell's response is to use local agents, close or co-located with the base stations, within the operator backhaul that inspect UE uplink and downlink traffic and extract information that allows that packet to be classified under one of these aggregation dimensions. For e.g., one agent will decapsulate the GTP header and extract the UE IMEI and use that to assign this packet to a flow rule that aggregates by UE Id (option c).

Table 2.5: A table of sample Service policies [31]

Prio	Predicates	Service Actions
1	provider = B	Firewall
2	provider != A	Drop
3	app = video ∧ plan = Silver	[Firewall, Transcoder]
4	app = VoIP	[Firewall, Echo-Cancel]
5	device type=M2M fleet	[HighPriority, Firewall]

To put SoftCell's policies into perspective, Table 2.5 outlines a set of sample policies for MNO-A. MNO-B has a roaming agreement with provide A but all that traffic needs to be passed through a firewall middlebox. The 2nd clause indicates that traffic from all other providers will be dropped. Clauses 3-5 will apply to MNO-A's traffic. Video traffic will go through a middlebox chain comprising of a Firewall and a Video Transcoder. Similarly VoIP traffic will go through a service chain consisting of a Firewall and Echo Cancelling middlebox. What can be seen here is that higher order matching conditions can be build by using the base IP tuple or L2 matching conditions.

SoftCell's vision of the modern day SDNMN is one which is comprised of a series of middleboxes which can be linked together to provide network services chaining. The term "boxes" may elude the reader to believe that these are dedicated hardware, however

SoftCell makes it clear that these middle boxes may be implemented as virtualized instances running on one or more hosts along the lines of NFV. To this end the "action" part of these aggregated flow rules will be a list of middleboxes to steer the packet through. The complexity lies with the underlying switches to work out the next hop(s) leading to the required middlebox and to maintain path states for these middleboxes.

In our perception, using middleboxes within the EPC core network appears to have a number of benefits. Among other things, we can argue that present day S-GW and P-GWs are too top heavy i.e. have functionality built in them that may not be needed for all sessions. Even more so is the fact that given the centralized nature of these gateways at the edge of the core EPC network, UE traffic may need to traverse large distances to use gateway services that may be better served closer to the Radio Access edge. Having middleboxes distributed across the core network will allow only pertinent services to be chained together instead of centralized gateways handling a number of vastly different requests.

The recent work done by Ko et al in [33] recognizes the merits of middleboxes as a means to decompose EPC core functionality into more granular service components that can be chained together. Their premise is the same as that of SoftCell i.e. not all UE sessions require all the functionality provided by the centralized gateways and some of these services are better suited closer to the RAN edge. Additionally however, Ko et al focused on the optimal placement of middleboxes within the CN by proposing a function that minimizes the average packet transmission cost across service chains and the increase in packet transmission delays across these chains after handovers. Their design requires modifications to the EPC bearer establishment function, where-in instead of selecting a S/P-GW pair, the MME will select a sequence Service Nodes (SN) that act as middleboxes chained together. Under Basta's terminology, their design can be considered as *fully decomposed* where both control and user plane functionality is moved into these middle boxes.

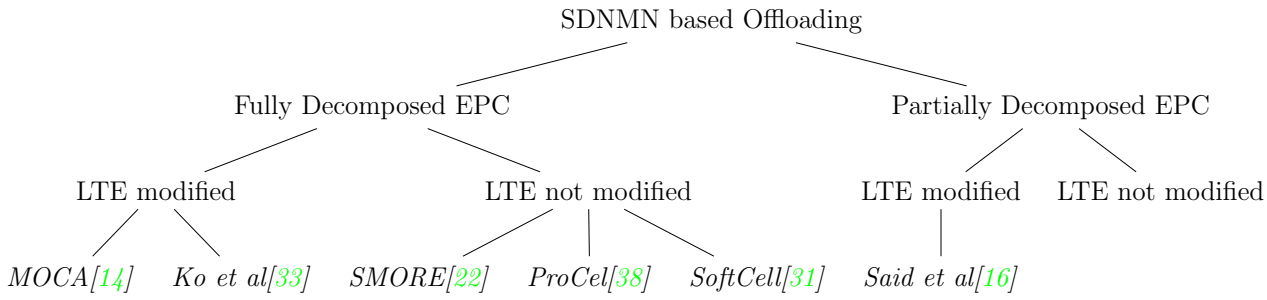

Figure 2.7: Taxonomy of surveyed SDNMN offloading techniques for core network traffic

The Taxonomy tree presented in Figure 2.7 depicts a summary view and classifies the works surveyed in this sections across two tiers a) whether full or partial virtualization of EPC core components is required, and b) whether a change in the LTE - EPC standard will be required. This depiction aims to draw a correlation between the different works in this chapter.

Chapter 3

Proposed Methodology

Our main contribution in this work is to derive a design that enables us to simulate and benchmark traffic offloading from the EPC core network to a fixed IP network driven by policies enforced by an SDN controller. In addition, we will explore the implications of this traffic offloading on the CN, and shall quantify boundary conditions under which offloading is feasible. Our design shall take inspiration from the works outlined in the the previous chapter.

The rest of this chapter is organized as follows: the first section will attempt to formally introduce the architecture of our proposed framework and its components. This is followed by a section that introduces the set of performance metrics that our proposed framework employs in order to gauge the effectiveness of offloading. Finally with foresight from these performance indicators we shall devise a mathematical relationship between the entities being offloaded and the optimal conditions for this offloading.

3.1 System Architecture

It is generally considered good design practice for the designer to declare, for lack of a better word, a set of *design non-negotiables*. We shall follow suit based on our analysis of the Related Works from which our design is influenced and can be considered as an amalgamation of:

- the design should discourage modifications to standard LTE components, interfaces and behavior. Any such changes would require buy-in from the 3GPP standardization body.
- benefits in the user plane, via offloading, should not be achieved on the back of increased complexity in the control plane. Existing offloading techniques such as LIPA, SIPTO and S1-Flex require additional signaling to set up the offloading pathways.
- the offloading network should complement the MNO's core network and not replace it.
- in the absence of inline GTP parsing on the OF switch, GTP header encapsulation/decapsulation is best served via purpose-defined middleboxes located at proximity to the RAN and not at the gateways at the core network edge. This is to minimize processing delays experienced by sessions which are sent to the GTP parsing middlebox, only to be sent back since they were not marked for offloading.

The next sub section will investigate these design considerations in-turn and emphasize how they influence our proposed framework. To ease the forthcoming discussion, we will here-on-after refer to our architecture as AETOS, short for **A**gnostic **E**pc **T**raffic **O**ffloading through **S**DN. The section that follows will outline AETOS's workflow and message primatives.

3.1.1 AETOS - Design Considerations

While standardization may stifle radical creativity, it is done for good reason: to promote vendor inter-operability and predictability of functionality. Offloading architectures, that modify LTE standards, such as those summarized in Figure 2.7, may bring the biggest benefits but will be met with stiff resistance from MNOs and 3GPP lobbyists. Our design framework needs to operate within the confines of the LTE standard.

One observation we eluded to earlier was the fact that existing LTE offloading mechanisms such as SIPTO and S1-Flex provide data plane alleviations at the expense of additional signaling required to establish those offloading pathways. To this end, UE agnostic

offloading solutions such as MOCA, SMORE and ProCel, are attractive design inspirations for our framework.

Finally for the design to make practical sense, from an implementation perspective, it will require modifications within the provider's backhaul network i.e. the series of switches and ethernet links that connect EPC components with each other in the same switching domain and/or the routers involved if the EPC components span multiple subnets or routing domains. Above all, the backhaul network must now comprise of *Openflow-hybrid* [24] switches that can talk upstream with an SDN controller. The Openflow switch must be able to steer traffic away from the core network (either the MME or the S-GW) to a fixed IP gateway that represents the offloading server. The backhaul network will also need to comprise of aggregation points, located near the RAN edge, and which terminate multiple enodeB's within that geographic region and connect them to the EPC network. Fortunately, such aggregation points already exist within the mobile backhaul and are referred to as Mobile Telephone Switching Offices (MTSO's) [19]. In our design, an OF switch, with middleboxes co-located at the switch, would be a perfect embodiment of an MTSO and represent an optimal placement for our offloading framework. Therefore these MTSOs shall act as traffic funnels either to and from the EPC or to other MTSOs.

Figure 3.1 illustrates the logical breakdown of an MTSO as it relates to our system architecture. Each MTSO will be comprised of an Openflow Switch, with a co-located middlebox that will be responsible for GTP header parsing and extraction. As can be seen in the figure, MTSO A serves as a middle-man between the EnodeB's in the radio access network, and the MME and S-P/GW nodes in the EPC. Such deployments will be hierarchical with MTSOs interconnected with each other, contingent on the size & complexity of the mobile operator's backhaul network.

The MTSO's endpoints or ports can be described as, in no particular order:

1. **EnodeBs:** The primary purpose of an MTSO is to serve as an aggregation point for multiple enodeBs so that they have a single pipe connection to the EPC, as opposed to a point-to-point connection model. The underlying switch within the MTSO will be responsible for the physical realization of the logical X2 interfaces between enodeBs

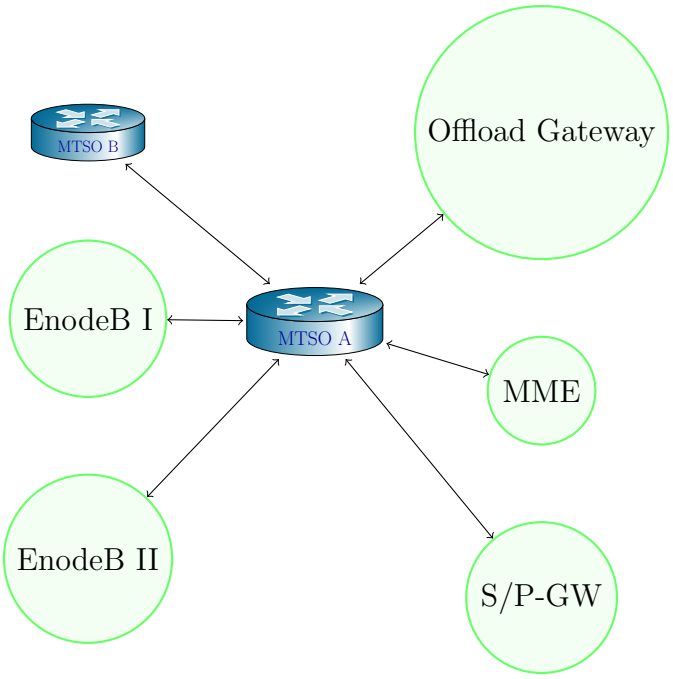

Figure 3.1: MTSO Deployment

(for intra-LTE handovers), as well as the enodeBs S1-MME and S1-U interfaces to the MME and S-GW respectively. All such traffic will travel the switch which will be responsible for forwarding it across the MTSO endpoints indicated in Figure 3.1

2. **S/P-GW:** This represents the standard EPC gateways which provide Internet and end-user services via PDN and IMS. Notice that neither PDN nor the IMS are endpoints to the MTSO. Reason being that in order for a component to be considered as a candidate for an MTSO switch port it has to have upstream interfaces to multiple other components, all of which are in turn switch ports on the MTSO. In the case of PDN and IMS, their only upstream interface is to the P-GW and can therefore be modeled in our framework as a point-to-point link.

3. **MME:** This represents the standard MME. The underlying switch within the MTSO is the physical embodiment of MME upstream interfaces such as the S1-MME and the S11 to the enodeBs and the S-GW respectively. In our design, HSS is connected to MME via a point-to-point link.

   Recall from our previous discussion that an MME may be pooled with other MMEs

to provide S1 flex load balancing technology to the enodeBs. While the illustration in Figure 3.1 only depicts a single MME endpoint, in reality there may be N of these connected to the MTSO; alternatively the MMEs may be pooled by interconnecting MTSOs each with their individual MME endpoint(s). The exact nature of this realization is left at the discretion of the MNO - the design however allows this flexibility.

4. **Offload Gateway:** The offloading gateway is the embodiment of the fixed IP network to which core network traffic will be steered. This can be either a cloud based EPC such as the hybrid design stipulated by Basta in [15] or the one in MOCA[14]. Alternatively it could be a simple internet gateway such as those in SMORE[22] and ProCel[38]. The onus is on the MNO to decide which type of offloading deployment works best for them.

While the MTSO is an essential cog in the machine that is AETOS, it is by no means a definitive representation of the framework. Within AETOS, an MTSO can be viewed as a data plane device that receives instructions from a centralized SDN controller. Figure 3.2 illustrates a system level view of AETOS's architecture and the components involved.

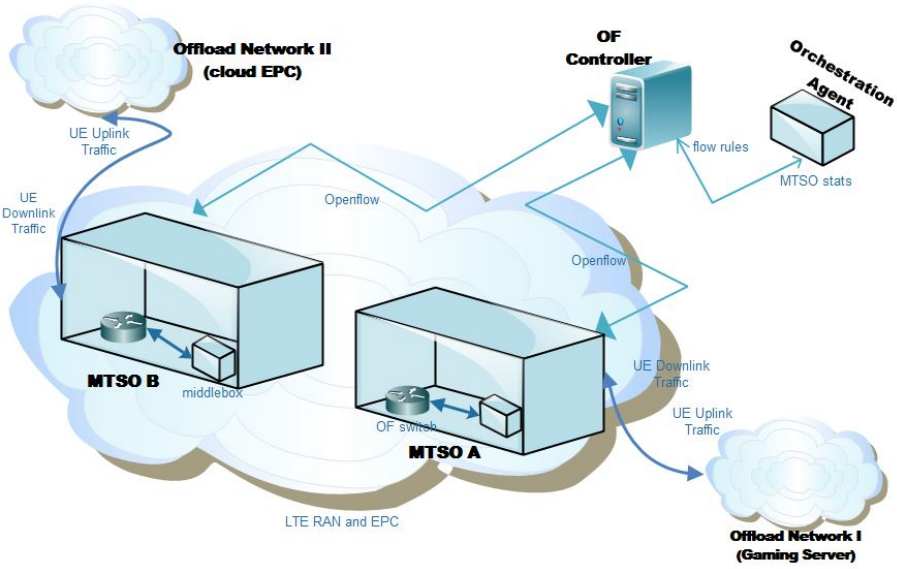

Figure 3.2: AETOS System Architecture

In this embodiment, a northbound controller talks Openflow with data-plane devices that have middleboxes co-located at the switch. Collectively, this synergy represents an MTSO entity such as the MTOS's *A* and *B* shown in the figure. The middlebox within each MTSO is responsible for a) GTP header parsing, and b) extracting and storing UE session information (such as TEIDs) for use during agnostic offloading.

Since the middlebox will need to recirculate packets back into the switch after parsing them, it cannot be connected as an MTSO endpoint (switch port). For the packets that are re-circulated back into the switch, the middlebox can further optimize the system by deciding which port to forward the packet to, instead of it going through the switch's regular forwarding table. For instance, packet A is received at the OF switch via its enodeB endpoint. The switch sends this packet to the middlebox for GTP parsing. The middlebox extracts the UE's IMSI from packet A and determines that sessions for this UE IMSI were marked for offloading. Now on injecting the packet back into the switch, the middlebox can *tell* it to use the offload network endpoint as the forwarding port. In the context of Openflow, it would seem like that the middlebox has to send a Packet-Out message to the switch that encapsulates packet A and sets *action: output to offload network endpoint*. For this reason, and other similarities to OF controllers, the middleboxes in AETOS will be considered as localized SDN controllers.

The Openflow standard allows for an OF switch to have multiple control channels. Since the middlebox will have to issue both read and write openflow commands, it will function as an EQUAL mode controller with respect to the global SDN controller.

As an additional optimization, the global SDN controller may dynamically deploy the middlebox as a Service Chain node. This saves the MNO from permanently reserving system resources, even during periods of low traffic, where offloading and therefore the middlebox may not necessarily be required.

The LTE RAN and EPC, components of which form the MTSO endpoints, have been abstracted in the drawing and are represented by an *LTE-EPC cloud*. AETOS supports multiple types of offloading networks as its endpoints such as *Offloading Network I and II* shown in the figure. Where-in the former is a full-fledged cloud based EPC, while the

latter is a simple gaming server providing end-user services to select UE sessions.

Another key component of the AETOS architecture is the *Orchestration Agent* which connects to the SDN Controller. This agent provides an Operations, Administration and Management (OAM) interface for application designers and OSS/BSS systems to send offloading commands to AETOS. These commands include, but will not be limited to, either UE IMEI information for specific UE sessions that need to be offloaded, or more generally EPS Bearer Types for a broad group of sessions that will be offloaded. The Orchestration Agent shall translate these instructions into specific flow rules that will be passed along to the controller, which in-turn will push them down to the data plane. In order to craft these flow rules, the agent will require access to UE session information, which the MTSOs will passively collect & extract from EPC control messages. These *MTSO Stats* will be passively provided to the Orchestration Agent via the OF controller; alternatively the Agent may relay a *Statistics Query* READ-STATE message to the data plane, in order to retrieve MTSO statistics on the fly.

Having exposed the reader to AETOS's core components, the next section shall focus on the interactions between said components and the semantics of these exchanges.

3.1.2 AETOS - Message Primitives and Workflow

AETOS has three levels of workflow and interactions:

1. MTSO $\leftrightarrows$ LTE-EPC interactions
2. SDN Controller $\leftrightarrows$ MTSO interactions
3. Orchestration Agent $\leftrightarrows$ SDN Controller interactions

Our investigation shall proceed in a bottom-up fashion, starting with the MTSO and working our way upwards.

3.1.2.1 Level 1 Interactions

As the "A" in the name suggests, AETOS is a UE agnostic offloading architecture. What this implies is that the UE has no knowledge that its sessions have been offloaded. The UE assumes all Uplink and Downlink traffic comes over the S-GW/P-GW to which it originally registered. The converse would have required AETOS to register the offloading gateway at the UE at the expense of an increase in signaling complexity at the RAN-EPC edge. Therefore to support agnostic offloading, AETOS needs to ensure the following:

- For **Uplink** replace the original *destination* endpoint with the offload gateway's endpoint
- For **Downlink** replace the *source* endpoint, which will be the offload gateway's endpoint, with the original S-GW endpoint

To this effect, the MTSO switches not only need to forward packets across their endpoints, but would also be required to re-write the packet headers so that the recipients can process them accordingly. Since traffic in the LTE core network goes over point-to-point GTP tunnels, this would entail rewriting the TEIDs. As SMORE[22] and MOCA[14] have already shown us all such information can be passively extracted from a *UE Attach* procedure within the LTE - EPC network.

Figure 3.3 depicts a call flow diagram for this procedure. The UE International Mobile Subscriber Identity (IMSI) is extracted from the UE's *Attach Request* primitive sent to the MME via the enodeB. The UE IP address, S-GW TEID and S-GW IP address will be extracted from the MME's *Attach Accept* response and eventually the enodeB IP address and TEID from its *Attach Complete* response.

These extracted attributes are stored in the middlebox, and is referred to as the *UE Extraction Table*. A snippet of which is illustrated in Table 3.1. The middlebox periodically sends these extractions upstream to the Orchestration agent which aggregates them from multiple MTSO sources and uses them to construct flow rules(include packet rewrite rules).

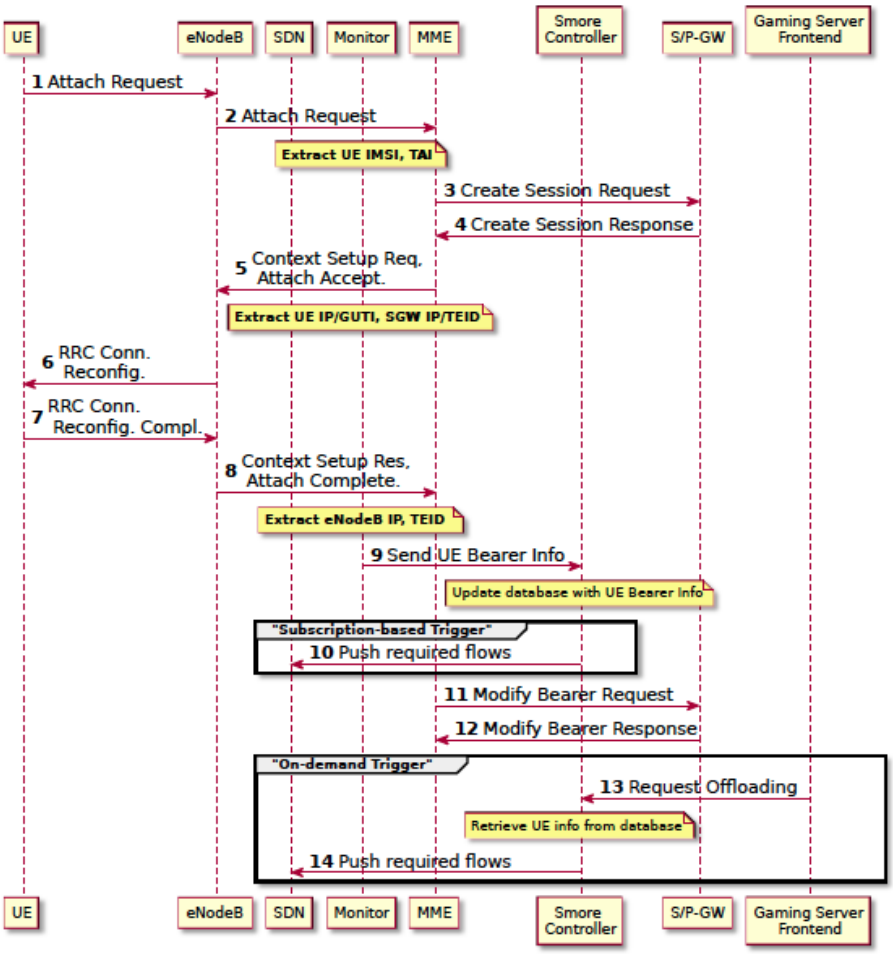

Figure 3.3: UE Attach call flow diagram [22]

UE IMSI	UE IP	S-GW TEID	S-GW IP	enodeB TEID	enode IP
310150123456788	1.1.1.1	1	192.168.1.1	100	10.10.10.1
310150123456710	2.2.2.2	2	192.168.1.2	200	10.10.10.2

Table 3.1: UE Extraction Table with sample entries

These mappings shall remain active till the the UE to enodeB and S-GW mapping remain the same. Therefore the architecture needs to account for X2 Handover scenarios where a UE may be handed off to another enodeB and the extracted attributes pertaining to that UE will have to be changed. Note that since AETOS only accounts for offloading within an MNO's home network, UE roaming scenarios, where a session is handed off to an external network, would cause the extracted attributes for that UE to be purged.

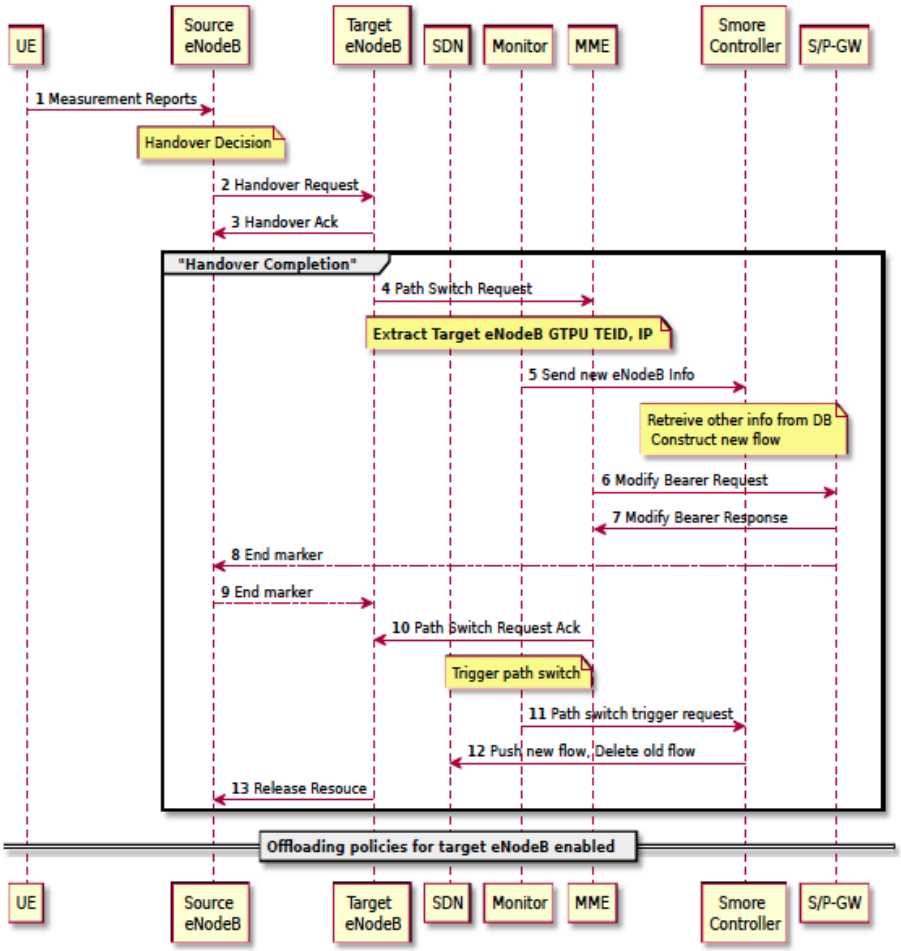

Figure 3.4: X2 Handover call flow diagram [22]

Figure 3.4 depicts the UML call diagram for an X2 procedure. Once handover is complete, the new enodeB will transmit a *Path Switch Request* to the MME. The new enodeB TEID and IP address shall be extracted from this message primitive and will update the extraction attributes for that UE. Now if both source and target enodeB's are on the same MTSO, as will likely be the case since an MTSO typically covers a large area, then this is a matter of simply updating that record on the MTSO middlebox. This update will be pushed upstream to the orchestration agent on the next push cycle. However if the target enodeB is on a different MTSO then the UE's extraction record will need to be forwarded to the new MTSO over the X2 interface.

3.1.2.2 Level 2 Interactions

To reiterate our previous discussion, the Openflow standard does not allow for inline GTP parsing. AETOS's solution to this limitation is to co-locate a GTP parsing middlebox with the OF switch which will function as a localized EQUAL mode controller. For the purpose of this discussion, we shall call this the *secondary controller*, with the global controller, that talks to multiple MTSOs, being referred to as the *primary controller*.

On receiving an offloading command, the Orchestration Agent will use the primary SDN controller to install a flow rule within the pertinent MTSO switch. Recall from our discussion in the Introductory chapter; this flow rule will be based on the IP five tuple attributes. As far as the primary SDN Controller is concerned, the switch needs to forward all packets between an enodeB $\leftrightarrows$ S-GW endpoint pair to the middlebox for further processing. Since flow rules are not bi-directional, the primary controller will therefore install two rules for Uplink and Downlink - reflected in Table 3.2 and Table 3.3

*	*	*	*	*	*	*	enodeB IP	S-GW IP	*	*	*	CONTR-OLLER
Ingress Port	meta data	src MAC	dst MAC	VLAN id	VLAN priority	MPLS tag	src IP	dst IP	IP ToS	L4 src port	L4 dst port	action

Table 3.2: Uplink rule for Offloading

*	*	*	*	*	*	*	offload IP	enodeB IP	*	*	offload Port	CONTR-OLLER
Ingress Port	meta data	src MAC	dst MAC	VLAN id	VLAN priority	MPLS tag	src IP	dst IP	IP ToS	L4 src port	L4 dst port	action

Table 3.3: Downlink rule for Offloading

The *action* in both these rules is to forward matching packets to the Controller for further processing as *Packet-In* messages. Normally, the OF switch would dispatch these messages to all active Control channels which means that both the primary and secondary controller will receive it. The primary controller in AETOS therefore needs to disregard these messages or may cache them for auditing.

The processing that happens at the middlebox can be summarized by the algorithm

given below:

Algorithm 1: Middlebox Packet Processing

Data: *Packet In* message

Result: *Packet Out* message

```
1  begin
2      strip outer most IP header
3      decapsulate GTP header
4      parse source and destination TEID and UE IMSI
5      if UE IMSI tagged for Offloading then
6          if matched Uplink rule then
7              if if offload gateway is S-GW then
8                  modify GTP header→destination TEID = offloadServer TEID
9              action_1 →Rewrite Destination IP(offload gateway IP)
10             action_2 →Output(offload gateway endpoint)
11             add action_1, action_2 to action_list
12         else if matched Downlink rule then
13             modify GTP header →source TEID = S-GW TEID
14             action_1 →Rewrite Source IP(S-GW IP)
15             add action_1 to action_list
16     else
17         action_1 →Output(PacketIn out port)
18         add action_1 to action_list
19     Create Packet Out msg with action→action_list
```

As a consequence to the forward-to-controller rules in the OF switch, the middlebox will now receive *Packet-In* messages. We benefit from an optimization that exists within the Openflow standard where the payload of the received Packet-In message is only the IP header and not the full original packet. A Buffer Id is provided within the message incase the controller requires the full packet.

The middlebox strips the outer header and separates the GTP header in order to extract

the endpoint TEIDs as well as the UE identifier (Steps 2 -4). Since all peer UE sessions, i.e. those that travelled the same enode↔S-GW as the offloading UE will also be forwarded to the middlebox, it will need to filter those out by walking the UE Extraction Table (outlined in Table 3.1) using the IMSI as the key (Step 5).

Now if this packet came in due to an Uplink rule (Steps 6-11), we know we need to steer this towards the offloading gateway. Therefore the destination IP would be rewritten to that of the offloading gateway. To the same tune, the Output switchport will need to be changed to the offloading gateway endpoint, from the original S-GW to which the packet would have been forwarded to had the middlebox not intervened. These fields will eventually be set as *SetNWDst* and *Output* actions respectively, in the *Packet-Out* message that the middlebox fabricates and sends down to the OF switch. If the offloading gateway is an S-GW (for the cloud based EPC scenario) then the destination TEID will also need to be rewritten. Since the Openflow standard does not allow for any OF actions to rewrite GTP headers, this rewrite will need to be done in the middlebox when creating the payload of the Packet Out message.

On the flip side, if this packet came in via a Downlink rule, and if is for an Offloading UE then we know that this packet originated at the offloading gateway. The source IP will need to be rewritten to indicate the original S-GW IP via a *SetNWSrc* action. Also the middlebox will need to rewrite the S-GW TEID to be that of S-GW.

Finally if the packet belongs to a UE that was not marked for offloading then it'll be pushed back to the switch as-is (Step 16)

3.1.2.3 Level 3 Interactions

At the top of the pyramid is the AETOS Orchestration Agent which is responsible for receiving offloading commands on its northbound interface and using those to orchestrate flow rules on its southbound interface to the primary controller. AETOS will support two types of offloading:

- **UE specific offloading:** UE IMSI provided in offloading command

- **Session based offloading:** Session QoS parameters, such as EPS Bearer Type and QoS Class Identifier (QCI), provided in offloading command

It is necessary to support both these scenarios as some deployments may demand redirecting all sessions belonging to a specific UE. An example would be a customer paying a premium for such as service, such as the real time gaming user outlined in SMORE and MOCA. Typically however multiple applications may be running on a UE, each one with its specific QoS requirements. MNOs deploying AETOS may take a page out of ProCel's book by enforcing offloading for all delay tolerant traffic. In this case we don't care about specific UEs but rather their QoS profiles.

Table 3.4 outlined the set of QCIs and their corresponding bearer types and descriptions as defined in the LTE standard. Going back to our "delay tolerant traffic" example, one command may prescribe offloading all traffic with $QCI \geq 5$. Another may specify offloading all MBR bearer traffic regardless of QCI. The onus is on the Orchestrator to recognize the command requested and translate it into one or more flow rules.

QCI	RESOURCE TYPE	PRIORITY	PACKET DELAY BUDGET (MS)	PACKET ERROR LOSS RATE	EXAMPLE SERVICES
1	GBR	2	100	10^{-2}	Conversational voice
2	GBR	4	150	10^{-3}	Conversational video (live streaming)
3	GBR	5	300	10^{-6}	Non-conversational video (buffered streaming)
4	GBR	3	50	10^{-3}	Real-time gaming
5	Non-GBR	1	100	10^{-6}	IMS signaling
6	Non-GBR	7	100	10^{-3}	Voice, video (live streaming), interactive gaming
7	Non-GBR	6	300	10^{-6}	Video (buffered streaming)
8	Non-GBR	8	300	10^{-6}	TCP-based (for example, WWW, e-mail), chat, FTP, p2p file sharing, progressive video and others
9	Non-GBR	9	300	10^{-6}	

Table 3.4: Standardized QoS policies within LTE [36]

The AETOS Orchestration Agent may support two broad categories of commands: i) **set** commands, received on its northbound interface and for which the response is an ACK/NACK or a Transaction Id, and ii) **get** commands used to issue a read operation on its southbound interface and for which the response is the data item requested.

Table 3.5 outlines a terse list of set/get commands that the AETOS Orchestrator needs to support and make available externally on its interfaces

Command	**Calling Entity**	**Arguments**	**Return Type**	**Description**
setOffloadingAdd	*external*	offload conditions	transaction id	offloading based on a list of one or more match conditions such as UE IMSI, QCI, EPS bearer type
setOffloadingRemove	*external*	transaction id	ACK/NACK	revert a previously applied offloading policy based on its transaction id
getOffloadingStats	*external*	transaction id & stats type	stats item	query command used to retrieve information of an offloading policy including the number of flow rules that was installed due to this policy
getMtsoStats	*orchestrator*	mtso id & stats type	stats item	query command used to retrieve a data item from a data plane MTSO device

Table 3.5: AETOS Orchestration Agent commands

On receiving a *setOffloadingAdd* command, the Orchestration Agent will parse the matching conditions and do lookups on its aggregated UE Extraction Table in order to construct the flow rules. For any information that the Agent needs and is missing in its Extraction Table, it will request it from the data plane by sending a *getMtsoStats* command. Once the Orchestrator passes the flow rules to the primary controller and

receives a success message indicating that they have been pushed down into the MTSOs, it'll construct a Transaction Id for this offloading policy and respond with it to the external caller. The obligation is on the caller to store these transaction Ids for future reference such as policy modification or deletion via the *setOffloadingRemove* command. Additionally, an external source may request detailed information about the Offloading policy that was applied such as the number of flow rules that resulted, openflow stats counters for number of matches to these flow rules, the duration for which the Offloading policy has been active etc. These functions shall be provided by the getOffloadingStats command.

It is noteworthy to mention here that the commands and interactions mentioned above either originate externally or at the orchestrator. However looking back at Figure 3.2, and the associated text that followed, we do know that the primary controller will periodically send the orchestrator asynchronous feedback messages from the MTSO data plane. These messages include:

- MTSO level OF flow counters and statistics
- A snapshot of the UE Extraction Table within this feedback cycle indicating new entries that have been added

While these messages may appear *asynchronous* from the Orchestrator's perspective, in reality they are a consequence of the primary controller polling the MTSO in its feedback cycle. The responses that the controller receives from the MTSO are prepended with the MTSO's control channel Id which we refer to as *mtso id* and will be stowed away by the Orchestrator for use if it ever wants to request information from this specific MTSO.

3.1.3 AETOS - Design Limitations and Constraints

Having explored the many cardinal features of AETOS in detail, let us take a step back and discuss some of the framework's design limitations.

1. **offloading UDP based sessions only :** UDP being a connection-less protocol allows for agnostic redirection. The same cannot be said for TCP based sessions

since both source and destination need to negotiate a session. Therefore even if UE flows were redirected to the offloading gateway, the TCP transport layer would drop the packet due to TCP checksum failures.

Conversely the offload gateway could not originate downlink packets for the UE without negotiating a 3 way handshake first at which point the framework is no longer **agnostic**. However all is not lost here: the applications for which AETOS is envisaged are data-intensive real time video(interactive and online) and gaming applications which typically use a UDP transport

2. **inter-LTE handovers:** AETOS does not have provisions for offloading roaming UE sessions within its home network. This would require coordination between the AETOS in the local network and the roaming UE's HSSs and for the AETOS Orchestration agent to be able to access UE subscriber information. The situation becomes increasingly complex when the visiting network does not have an AETOS deployment and therefore some mechanism would be required for AETOS in the home network to deduct the AETOS capability of the visiting network.

   It must be noted here that while the first limitation, in this section above, arises simply from the nature of the framework, this limitation results from a lack of fully understanding the complexity involved in supporting inter-LTE handovers.

3.2 Performance Metrics and Other Measurements

Up until this point, we have investigated the AETOS architecture in a fair amount of detail. However, AETOS needs to be scrutinized to gauge its practical viability. For one, AETOS has specifically shied away from any changes in the LTE standard simply because, in our perception, it is a path of maximum resistance. We also remain cognizant to the fact that improved operating efficiency in the LTE core network, via traffic offloading will incur a performance hit in order to steer flows towards the offloading network. We are also aware of the practical limitations of OF switches vis-a-vis inline GTP parsing and the penalty incurred in sending all its flows to the middlebox.

We therefore stipulate the following measurement goals for any prototype deployment that aims to qualify the AETOS architecture:

1. **The Round-Trip-Time(RTT) must not exceed the assigned packet delay budget of the UE session:** Intrinsic to a UE session's QCI profile is the acceptable RTT. For instance, for interactive video and certain online gaming applications, this is typically set at 100 - 150ms[36].

   Measurements will need to be taken to ensure that both the offloaded sessions, and the ones that were sent to the middlebox but returned (since they were not marked for offloading) both satisfy their RTT requirements.

2. **Based on the cardinal principles of seamless LTE handovers, offloaded UE sessions should not be disrupted during X2 handovers:** Measurements will need to be taken to ensure that AETOS can recognize a handover and update the UE extraction table and forward these updates across the X2 interface, all the while ensuring that the session is not disrupted.

3.2.1 RTT measurements

To validate the first goal, AETOS will need to employ throughput and packet delay measurement sensors at the MTSO endpoints. Since the discussion is focused on UE session RTTs, we will only consider UE initiated flows(Uplink). The RTT may be be computed as follows:

$$RTT_{uplink} = 2 * t_{UEtoEnodeB} + \delta t_{MTSOtoEnodeB} \tag{3.1}$$

Since AETOS is not a RAN level solution, it cannot independently measure UE to EnodeB transmission times, and therefore needs to either treat it as a fixed constant (based on MNO advertised capabilities) or determine its value via enodeB measurement reports. This transmission delay is represented by $t_{UEtoEnodeB}$ in the equation above, and needs to be counted twice for both outgoing and incoming directions. The measurement sensor at the enodeB endpoint will compute the time it takes for a UE flow packet to cross its endpoint and to be responded by with a corresponding downstream packet. The time differences

between the Upstream and Downstream packets shall represent the transmission delay through the EPC CN or the Offloading Network. This is represented by $\delta t_{MTSOtoEnodeB}$ in the equation above.

However a single RTT computation only represents a single point of interaction. Even for the same UE session traversing a given MTSO, RTT values may vary vastly depending on a) transient network conditions such as flux, b) packet jitter, c) instrument errors. This problem is compounded by the fact that the RTT criteria must be satisfied for all UE sessions in all AETOS MTSOs. Therefore to aggregate RTT values across different measurement iterations and locations, we shall model them as a Cumulative Distribution Function (CDF) curve for each QCI value. In doing so, multiple UE sessions having the same QCI value, and therefore the same packet delay budgets, will be modeled under the same CDF curve. If the RTT value corresponding to a CDF value in the 4th percentile (97% probability) is higher than the packet delay budget for that QCI, then we can state with a high degree of confidence that the offloading provided by AETOS violates the QoS impositions of one or more UE sessions.

3.2.2 X2 Handover measurements

Moving onto the second goal, it is evident that offloaded UE sessions will incur an additional step when being handed off from their source EnodeB to a target EnodeB, i.e. the MTSO middlebox shall need to update the UE entry in its UE Extraction table to stub out the source EnodeB attributes (such as IP address and TEID) with that of the target enodeB. When both source and target enodeBs are on the same MTSO, as will be the normal case, this is a simple search-and-replace operation.

However the situation becomes increasingly complex if the target enodeB is managed by a separate MTSO in which case the original MTSO needs to send this UE extraction table entry (with EnodeB attributes) to the target MTSO, and subsequently purge that entry from its own Extraction table. This creates an artificial delay in the Handover operation. The likelihood of failures is not while handover is in progress as all offloaded UE session traffic will be transferred seamlessly across the X2 interface, however as soon

as the target enodeB sends a *Path Switch Request* to the MME, we run the risk of having a stale entry in the Extraction table so that once Handover completes, Upstream packets for an ongoing offloading session may be sent to the S-GW instead of the offload gateway endpoint(since the MTSO may not have any rule for this new target enodeB). Even worse is the fact that these delays may trip off the *UE inactivity timer* and trigger an S1 release for the UE session either by the target enodeB or the MME causing the UE to transition to $ECM-IDLE$ mode [26].

The measurement sensor in this case, will therefore need to sniff the endpoints for certain control plane messages indicating a successful handover for an offloading UE. In addition to the Path Switch Request message that the MTSO middlebox already extracts, it will need to be instructed to keep an eye out for either a *UE Context Release* request from the target enodeB to the MME or a *S1 Bearer Release* command from the MME to the S-GW depending on whether the $ECM-IDLE$ transition is triggered by the enodeB or the MME.

3.3 Fairness of Proposed Methodology

The measurement goals outlined in the previous section serve as a basic sanity for qualifying AETOS as being field acceptable. However a design being practical does not imply that it is *fair*. For one, user sessions that had to be diverted to the co-located middlebox, only to be returned back can be deemed to have been treated *unfairly* if we contrast this to offloading sessions that benefited from a net improvement in RTT after being transferred to the Offload Network.

In the classical sense of the term, a design is deemed fair if every user determines that it has received the same equal portion of a shared resource [30]. In the context of this discussion, let us now assume that the "shared resource" is RTT. Each user, by virtue of the network, is provided an RTT. This RTT value is dependent on the QCI value of the session or the network path undertaken. We also argue that for MNOs to justify offloading, the RTT of offloaded sessions needs to be lower than that of CN sessions. MNOs can achieve

lower RTTs in the offload network(s), simply by virtue of it being a fixed IP network, being closer to the RAN edge, and having a smaller user base. Therefore right off the bat, we know that any measure of fairness would be tilted in favor of the offloading sessions. However the bigger question is what degree of unfairness can MNOs consider acceptable in lieu of improved operation efficiency. Let us now compute a fairness relationship between the user sessions in an AETOS enriched core network:

Let us assume that n UE sessions are active through an MTSO, let x be a measure of RTT with x_i being the RTT of the i^{th} session. Out of these n sessions, let us assume that r sessions are regular and not marked for offloading. All others have a constant lower RTT value of $x_offfload$. The JFI can then be computed as:

$$f(x) = \frac{(\sum_{i=1}^{r} x_i + \sum_{i=r}^{n} x_{offload})^2}{n(\sum_{i=1}^{r} (x_i)^2 + \sum_{i=r}^{n} (x_{offload})^2)}$$

The constant summation of the offload RTT can be simplified as:

$$f(x) = \frac{(\sum_{i=1}^{r} x_i + (n-r)(x_{offload}))^2}{n(\sum_{i=1}^{r} (x_i)^2 + (n-r)(x_{offload})^2)}$$

The quantity $(n-r)$ represents the number of sessions that will be offloaded. Practicality dictates that MNOs will not maintain different counts of offloaded sessions per MTSO, and would rather have a global *offload factor* that can apply to a number of MTSOs. Ergo, let us now re-write the $(n-r)$ quantity as a factor of r, represented below by γ:

$$f(x) = \frac{(\sum_{i=1}^{r} x_i + (\gamma \mathrm{r})(x_{offload}))^2}{n(\sum_{i=1}^{r} (x_i)^2 + (\gamma \mathrm{r})(x_{offload})^2)} \tag{3.2}$$

To simplify this relation further, let us now assume that all the regular UE sessions have the same constant RTT value (a more advanced adaption would model this RTT as a distribution), referred to as $x_{regular}$. The JFI value now becomes:

$$f(x) = \frac{(\mathrm{r} x_{regular} + (\gamma \mathrm{r})(x_{offload}))^2}{n(\mathrm{r}(x_{regular})^2 + (\gamma \mathrm{r})(x_{offload})^2)}$$

By expressing n, the total number of sessions as a factor of r, this relation can be further simplified as:

$$f(x) = 1/(\gamma + 1) * \frac{(x_{regular} + \gamma x_{offload})^2}{x_{regular}^2 + \gamma x_{offload}^2} \tag{3.3}$$

Based on the principals of polynomial expansion, we know that the numerator term will be bigger than the denominator and their ratio will be greater than 1. We also know that the JFI is bounded by a maximum value of 1. Therefore the possible range of JFI for the above set of constraints and assumptions now becomes:

$$\boxed{1/(\gamma+1) \leq f(x) \leq 1; \quad \forall x \geq 0} \tag{3.4}$$

MNOs will be more concerned with the lower range as it represents the worst case scenario. As an example, consider a deployment where 50% of the sessions are offloaded. Based on the relationship above, the system has a worst case fairness value of 0.67 meaning that 33% of sessions have been discriminated. Consider now the scenario outlined in ProCel [38] which postulates that 75% of traffic is "delay-tolerant" and therefore subject to offloading. The minimum JFI value now becomes 0.57 with 43% of UE sessions being treated unfairly by the system.

It is our hope that the relationship, derived above, serve as a useful measure for MNOs in deciding the degree of offloading in their networks, and to what extent a system bias can be tolerated in their deployments.

Chapter 4

Implementation and Evaluation

Thus far, we have explored the general idea of traffic offloading and optimizations in the LTE core network. Our study is based on some recent advances within the networking world such as SDN and NFV. We then highlight related academic work that leverages on these advances. Later, we discuss our own proposed solution along with its details and performance metrics. We couple that with a theoretical assessment of our model and derive a fairness relationship aimed at assisting MNOs in making well informed offloading decisions. Consequently, in this chapter, the focus will be on an experimental setup, and the insights drawn from this prototype deployment.

4.1 Implementation

To implement an AETOS prototype, we are presented with two possible options:

1. an **emulation** testbed
2. a **simulation** testbed

Emulation will involve studying the real-time interactions between the various components of AETOS. It is for this reason that an emulation testbeds will contain both hardware components, such as mobile devices and femto cells to model the RAN, as well as software

components, such as OpenEPC[25]. The primary advantage of using a network emulator as opposed to a simulator is that an emulation environment affords much higher fidelity, and accuracy[20].

On the contrary, simulation is done almost entirely in software. Simulation will allow AETOS designers to rapidly prototype and functionally verify their deployments without investing in specialized hardware. Additionally, while an emulation would use a real time clock (RTC), simulation models typically maintain their own simulation clock. This allows researchers to speed up and examine events that would otherwise take days or months to become observable. However it is important to note that undertaking an AETOS simulation does not diminish the need for an emulation. Together they complement each other to deliver benefits that either one alone cannot provide [1]; ergo they will be used in conjunction during the AETOS design validation process.

In this thesis, we shall consider a simulation model as being the first step towards formally validating the AETOS design. In the interest of time, it was decided to use an existing simulator, and enhance it to fit our need, as opposed to building one from scratch. Our choice of simulator was strongly governed by how well it supported both conventional LTE models as well as SDN/Openflow models.

After a brief survey, it was decided to move forward with *Network Simulator 3* (NS3) [17, 41]. NS3 is a discrete event simulator whose modular and open source nature makes it fairly popular in the research community. In the context of our deployment, NS3 has a near standard LTE RAN and EPC implementation, collectively known as the LTE-EPC Network simulAtor (LENA) project [12, 13]. The LTE implementation in LENA is based on 3GPP Releases 8/9 with some LTE-Advanced(Release 10) RAN features added after. This is sufficient for the purpose of our testing since the benefits brought by LTE-Advanced are concentrated mostly in the RAN, to which AETOS does not bear influence. On the other hand, the Openflow protocol in NS3 [6] is based on *version 0.89*, i.e. McKeown et al's original position paper on Openflow [37]. While the NS3 implementation does contain all the $OFswitch - controller$ message primitives that AETOS needs, one major caveat of that Openflow release is the lack of support for EQUAL or SLAVE mode controllers.

NS3's Openflow implementation is based on an external Openflow library (OFSID). An effort is underway to upgrade OFSID to Openflow *version 1.3* [5], which will in turn bring in support for multi-mode OF controllers, however at the time of writing this document that work is still not complete and the code is not available for public use. While AETOS does not specifically bank itself on multi-controller functionality, it does require the MTSO to maintain two concurrent control channels - one to the primary controller and the second to the middlebox based secondary controller. The present NS3 implementation of SDN co-locates a controller at each OF switch. In that sense, the controller in NS3 is synonymous (and shall represent) to the middlebox in the AETOS reference design. In light of this restriction, the scope of the experimental setup will need to be constrained to exclude the primary controller.

The next section shall explore said setup in detail and outline some of the implementation inhibitions that our choice of simulator poses.

4.2 Experimental Setup

Our experimental setup is an AETOS empowered LTE-EPC topology, based entirely in NS3. It must be noted that NS3 is by no means the epitome for the perfect test environment, specially in its SDN implementation. Mininet[35] is a popular SDN simulator with support for the latest Openflow standard as well as connectors to commercial controllers such as Open Daylight[4] and Floodlight[7]. However our work requires a simulator which implements both SDN and LTE functionality and which a dedicated test environment, such as Mininet cannot provide.

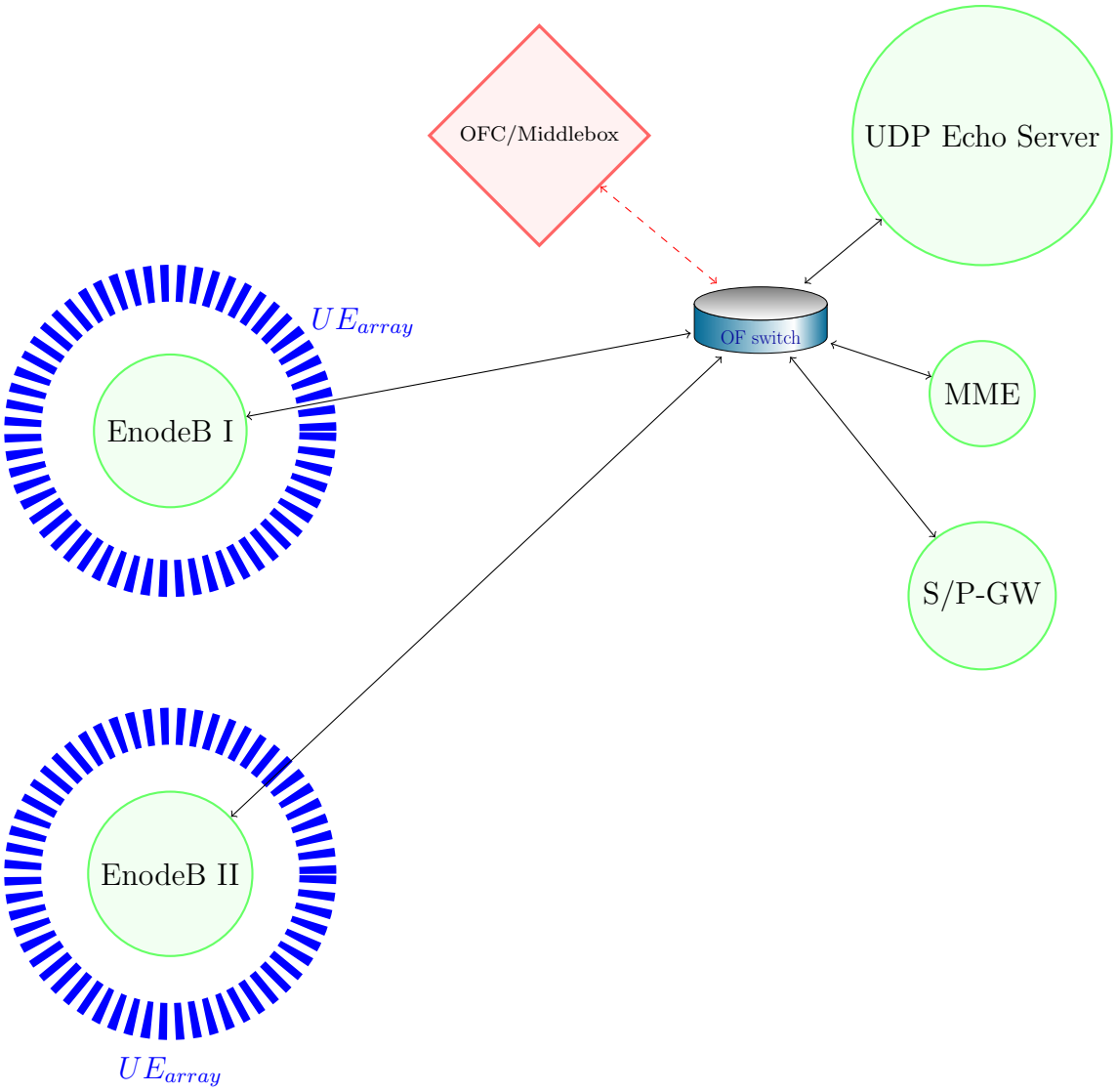

Figure 4.1: AETOS simulation testbed in NS3

Figure 4.1 illustrates the prototype deployment. The NS3 Openflow switch instance communicates with the co-located controller via its internal interface (not a switch port). Collectively these two components represent the MTSO whose end points are connected to S/P-GW, MME and EnodeB NS3 objects.

Recall, that MNOs have the freedom to configure their offloading gateways either as a fixed IP server (such as the gaming server example in MOCA/SMORE) or as a cloud based S/P-GW (Basta). NS3 has a limitation in the fact that only one MME and S/P-GW instance can be initialized over an EPC simulation. To support multiple such nodes would

require node attributes and configuration variables to be moved from a global instantiation to per class objects. Doing so is beyond the scope of this thesis as the aim is to test interactions over the AETOS fabric and not the processing that happens at end points. Therefore as a means of simplification, the offloading gateway is modeled as a UDP echo server, that echoes, in the Downlink direction, the payload of all Uplink packets that the MTSO forwards it.

Furthermore, the UE Extraction table, mentioned first in Section 3.1.2.1, is modeled as an SQLite [40] database which we have embedded inside the NS3 controller. To measure key packet metrics such as jitter, delay and size, we shall employ NS3 Flow Monitors[18] at critical nodes within the setup, such as the enodeBs and the offloading gateway.

As far as the LTE RAN is concerned, NS3 supports multiple pathloss and propagation models including support for urban obstacle structures such as Buildings however since AETOS is primarily an EPC level solution, it must remain invariant to RAN effects. For this reason, the UEs are placed equidistant to their enodeB in our NS3 test bed. These are shown as the UE_{array} blue rings in the figure. The default NS3 path loss model is used for the air interface between the UEs and their enodeB which is derived from the Friis Transmission Equation [29]

Additionally, recall that ProCel postulates that about 70% of sessions are delay tolerant. This is corroborated further by the independent study in [32] which states that 47.71% of the Internet usage is for HTTP applications and therefore susceptible to longer delays. Taking the middle ground here, we shall define and simulate 50% of the NS3 UEs as non-GBR bearers with longer RTTs and packet delay budgets. If all of these non-GBR UEs are offloaded then the system will have a theoretically minimum JFI value of 67% as prescribed in Section 3.3.

4.3 Evaluation

Our aim with evaluating the AETOS reference implementation is simply to declare, with degree of confidence, that the goals outlined in Section 3.2 are satisfied. This outcome

should not be taken as definitive proof but should set the stage, and serve as a precursor to further elaborate testing and prototyping. We begin with an evaluation of UE specific, and session specific offload behavior.

4.3.1 RTT and Packet Delay Budget Assessment

Recall that in our experimental setup, upto 50% of the UEs are marked as non-GBR bearers. The goal of this evaluation is two-prong, a) to offload specific UE sessions (across both EnodeBs), and b) to offload all non-GBR bearers. In both these tests, we will aim to categorize the net changes in RTT not only for the sessions that got offloaded but also for those that didn't.

The experimental setup shall scale UEs per enodeB in the following way: {2, 4, 8, 16, 32, 64, 128, 150}. Based on our simulations, the NS3 switch cannot reliably handle more than 150 UEs per enodeB, i.e. 300 active sessions traversing through it, as it results in a majority of the flows being dropped. The NS3 switch allows a number of packet queuing schemes, for simplicity we have adopted the *DropTailQueue* – a FIFO queue that drops tail-end packets on overflow [2]. Scaling the switch queue to maximum value, to accommodate more than 300 UE sessions, results in a sharp increase in RTT as well as intermittent SIGBUS crashes in NS3. It must be mentioned that the default NS3 LTE models don't suffer from these scalability limitations as they have the enodeBs, MME and PGW connected by dedicated point-to-point links, with no switching fabric in between(as is the case for AETOS).

The NS3 Flow Monitor captures, *RX Throughput*, *Packet Delay* and *Packet Jitter*. Table 4.1 summarizes these metrics for a baseline run that contains no offloading

Table 4.1: Baseline results for packet metric vs. UEs/EnodeB

	2 UEs	**4 UEs**	**8 UEs**	**16 UEs**	**32 UEs**	**64 UEs**	**128 UEs**	**150 UEs**
Tx/Rx Throughput (kbit/s)	224/112	495/248	871/277	784/278	955/252	1902/337	2995/368	4988/427
Mean Delay (ms)	30	57	871	792	936	1305	1386	1666
Max/Mean Jitter (ms)	35/8.6	71/23.5	122/28.6	251.5/28.2	318/28	650/20.4	637/19.7	1523/42

The values of throughput, packet jitter and delays are averaged over three consecutive

runs of each simulation. Similarly since both EnodeBs in our setup are equidistant from the OF switch(the MTSO) and therefore have the same path loss and propagation delays, we can compute a single average for these metrics over both EnodeBs. Jitter is another important metric for us as it represents the variation in packet delay between consecutive packets. Since GBR and non-GBR sessions are interleaved with each other, Maximum jitter represents the numerical difference between the best case packet performance (for a GBR session) and the worst case performance(for a non-GBR session). The maximum jitter values seen across both EnodeBs is averaged across three consecutive test iterations for each setup.

Evidently, the absolute values of these metrics do not reflect real-world behavior. Modern day switches and Openflow data planes are capable of processing 10Gbps+ of ingress traffic at line-rate which the rudimentary, single threaded implementation in NS3 simply cannot provide. Additionally, half the UE sessions were declared as GBR bearers and have a 100ms maximum packet latency requirement. While the LTE MAC scheduler in NS3 does prioritize GBR EPS bearer traffic over others and their P-GW implementation has priority queueing provisions, the NS3 OF switch itself is single queued with no notion of QoS paradigms, one would expect of a real switch, such as ingress policing/queueing and egress traffic shaping. Recall our earlier conversation where we postulated that any prototype that aims to qualify the AETOS framework may plot a CDF curve for each of the QCI values, represented within their setup. If the crown of the curve was lower than the max packet latency attribute of that QCI then end-to-end QoS session requirements would be satisfied. Since the NS3 implementation does not represent real-world deployments, but is simply a proof-of-concept, using such as CDF strategy would be a moot point.

However, not to paint a gloomy picture, the NS3 simulation does provide some key observations related to trend and how these values vary with scale. Insights drawn are as follows:

1. The Tx throughput (as seen at the enodeBs) increases in direct proportion to the number of UE sessions. However the Rx throughput eventually plateaus since the switch serves as a major bottleneck. Beyond this point any further increase in Tx

throughput translates into an increase in packet delay

2. The Average packet delay rises sharply when scaling UE sessions. Such an exponential rise may be observed in a real implementation as well and constraints the number of GBR sessions that an MTSO deployment can handle

3. The Maximum jitter represents the maximum bias between GBR and non-GBR sessions. Since the P-GW favors the former, non-GBR sessions get the short end of the stick and face longer packet queueing delays. It is for this reason also that a majority of the packet drops observed in this deployment are for non-GBR sessions.

   While the maximum jitter grows with the number of GBR sessions, its mean value paints a different picture altogether. Over multiple consecutive runs, it was observed that the mean jitter initially rises with the number of UE sessions, however adding more sessions causes it to reach an inflection point. This inflection represents the fact that a majority of remaining sessions are predominantly GBR while non-GBR sessions are now being dropped. The dropping mean jitter value is due to a increasing larger proportion of these GBR sessions in the running system. The non-GBR sessions that remain experience longer queueing delays thus accounting for the higher maximum jitter. We observed that the mean jitter will eventually spike again as the scale is further increased indicating that even GBR sessions are now being dropped and the remaining non-GBR sessions experience larger than ever delays - both factors contributing to the rising mean.

The experimental setup is now jigged to offload one specific UE session per enodeB and then subsequently to offload all active non-GBR sessions as postulated earlier. Figure 4.2 illustrates the variation of mean packet delay with an increasing number of UEs per EnodeB. A logarithmic curve of best fit is used to indicate the underlying trend as well as facilitate easier comparison between the three test schemes. Figure 4.3 outlines the variation in mean packet jitter against number of UEs/EnodeB. Given the cyclic nature of this trend, a fourth order polynomial is used as curve of best fit. The next subsections discuss these results

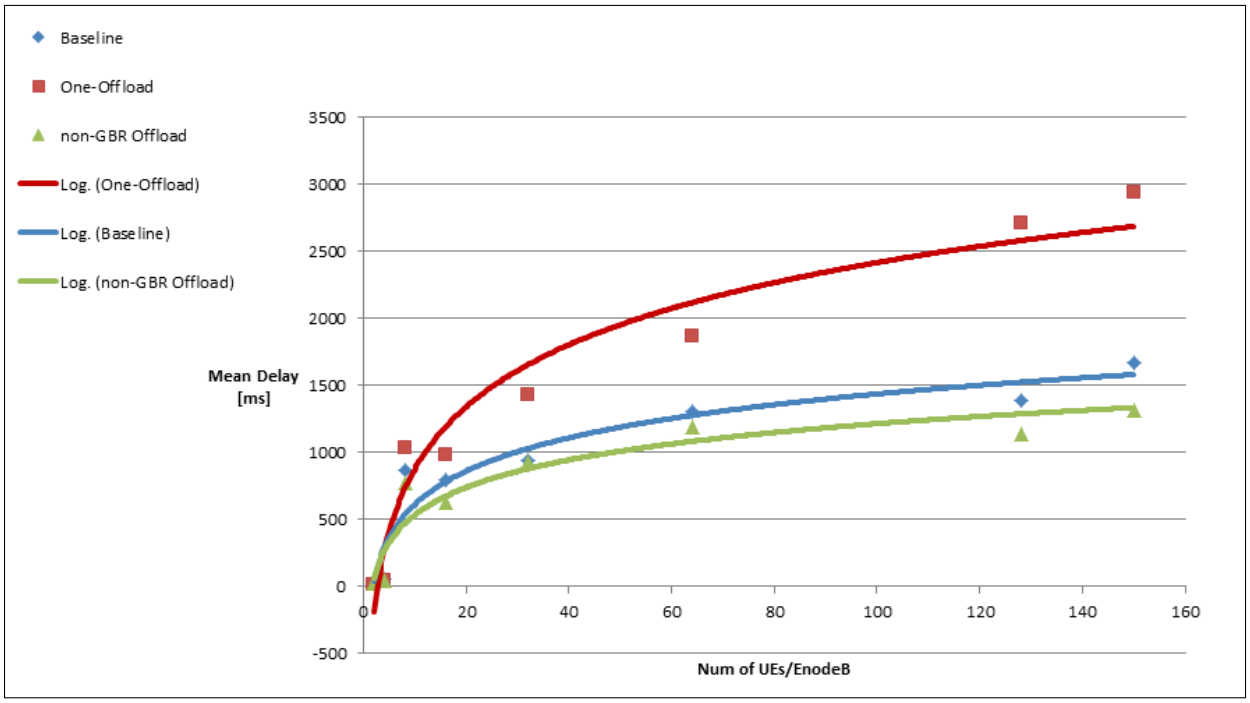

Figure 4.2: Variation in Mean Delay with UE scale over 3 different deployment scenarios

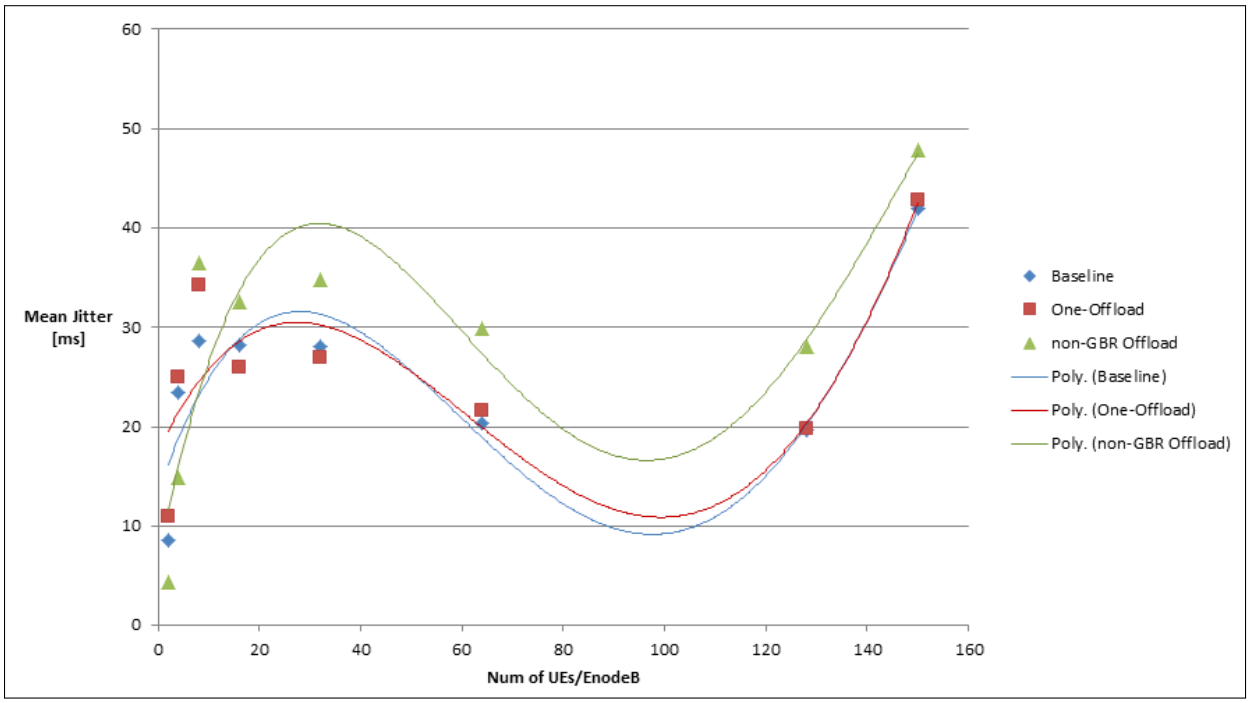

Figure 4.3: Variation in Mean Jitter with UE scale over 3 different deployment scenarios

4.3.2 One−Offload *vs.* Baseline

It is observed that when a single UE per EnodeB, is offloaded, all UE traffic between those EnodeB and the S-GW endpoint(s) is forwarded to the middlebox controller for GTP parsing. Most of these will be sent back to the switch, as-is, via *Packet-Out* messages. Those that do match the offloading criteria will experience must lower RTTs (owing to the simplistic nature of the UDP Echo Server that acts as the offloading gateway). It must be noted that this resonates with real-world expectation as the offloading gateways will be providing MNOs significantly lower RTTs than the core network. With regards to the one-offload scheme, small UE values upto four UEs per EnodeB experience a net reduction in mean packet delay since the scale-down in RTT via offloading exceeds the additional processing delay by moving these sessions to the controller. However as the number of UEs grow, the controller starts to be the bottleneck and performance declines exponentially, so much so that at 150 UEs/EnodeB, the observed mean packet delay is many folds higher than the baseline delay. The mean jitter represents the variation in packet delay between packets. The jitter value between two consecutive packets is maximized if one of them goes through the conventional core network pathway, while the other goes through the much fast offloading pathway. As can be seen in Figure 4.3, for small values of UEs/EnodeB, the mean jitter is higher than the corresponding value for the Baseline scenario. This is because, at smaller UEs scales, the proportion of offloaded sessions is high(50% for 2 UEs/EnodeB; 25% for 4 UEs/EnodeB) enough to cause the average jitter to rise. However this effect diminishes as the UE scale grows and the One-Offload curve appears identical to the Baseline scenario.

4.3.3 non−GBR Offload *vs.* Baseline

On the flip side, when we offload all of the non-GBR sessions, there is a net lower contention at the EnodeB MAC scheduling layer (for downlink) or the P-GW (for uplink). These layers can now focus better on prioritizing GBR sessions. It is for this reason that the mean packet delay improves significantly in this scheme and the benefit grows further with the number of active UEs, regardless of the increasing performance bottleneck at the

controller. Consequently also, the mean jitter in this scheme, as seen in Figure 4.3, is much higher than the Baseline curve. As more and more sessions are offloaded, the net difference between those that go through the conventional core network and those that go through the faster offloading pathway, grows and causes the mean jitter to swing in that direction. This serves as further testament to the fact that at larger scales, a dedicated and service-centric offloading server is better suited to performance than a centralized S-GW/P-GW service located far from the RAN edge.

These findings also resonate with what we postulated earlier: ***agnostic offloading in the face of non-inline GTP parsing only makes sense if a broad group of sessions are offloaded simultaneously***. Our analysis serves to bolster ProCel's delay-tolerant traffic offloading scheme while reminding us that the on-demand gaming server application devised by SMORE and MOCA will suffer from tremendous practical barriers.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

The work described in this thesis has been concerned with the general issue of traffic offloading within LTE core networks. With consumers becoming more and more data hungry, existing networks cannot scale dynamically to meet realtime capacity requirements. Fortunately, the telco industry is facing a pivotal moment in its history where concepts such as SDN/Openflow and NFV are bringing forth rapid changes within the network.

In this thesis we explored some of these concepts. We investigated works that leverage these next-gen ideas to provide novel traffic offloading schemes and compared them to existing offloading mechanisms such as LIPA, SIPTO and S1-Flex. When analyzing some of these works, we remain heavily biased in favor of scenarios that do not provide data plane offloading at the expense of increased signaling in the control plane. We therefore focus on the genre of *agnostic* traffic offloading where the endpoints remain oblivious to the presence of the middleware fabric that offloads its traffic. The perceived benefit of such schemes is the facility for mobile operators to rapidly and dynamically deploy service oriented gateways closer to the RAN edge. To this end, we cited a major limitation within the Openflow standard vis a vis its inability to parse LTE packet GTP headers inline at the data plane. Our work goes a step further and proposes using middleboxes and service chains as a compromise but provides fair warning on the severe implications of this

limitation in terms of the freedom and viability it affords to use cases such as *UE specific offloading*.

This discussion paves the way for AETOS - our traffic offloading design reference. Our work investigates the various components of AETOS such as the Orchestration Agent and its Primary and Secondary controllers, as well as the different layers of interactions intrinsic to this framework. We then derived a Jain's Fairness Index relation for our AETOS fabric with the aim of providing MNOs the ability to gauge the fairness of their deployment scenario in realtime based on variations in their network RTTs and user sessions. This work hopes that this mathematical relationship will serve further to incentivize offloading by allowing MNOs to justify their choices from a quantitative point of view, i.e. *how many sessions to offload given my fairness threshold?*, as opposed to a purely qualitative point of view, i.e. *should I offload or not?*. By asking the right questions and providing the right answers, mobile operators may gain easier buy-in from their stakeholders in rolling out such schemes.

This thesis then outlines an implementation strategy and highlights measurement goals that would be vital to qualify an AETOS deployment as being pragmatic. Our work then explores some of these cardinal principles by prototyping a simple AETOS on the NS3 simulator. Our findings further reinforce our perceptions; that such a fabric is best suited for scenarios where a broad group of sessions are offloaded together as opposed to individual sessions. This realization helps solidify our opinion against some of the related works in this area such as SMORE and MOCA while giving a thumbs-up to others like ProCel.

5.2 Future Work

The implementation done in this thesis is a simple proof-of-concept and may be developed further in a number of ways. We list some of these in increasing order of complexity.

5.2.1 Assessment of session disruption for Offloaded UEs in the face of X2 handovers

This reflects one of the Performance Measurement goals outlined in Section 3.2. A future work item may be to qualify this goal on a high fidelity LTE Emulator, one that is using real-time values for the *UE Inactivity Timer* or a real-world AETOS deployment.

5.2.2 OpenFlow v1.3 implementation in NS3

The Openflow implementation in NS3 is presently outdated and scheduled for an update in the 2016-2017 timeframe [5]. Once a stable distribution is available, our AETOS implementation in NS3 may be migrated to Openflow v1.3. One key advantage of doing so is the ability to bring in the Primary AETOS controller since Openflow v1.3 supports multi-mode controllers. The implementation should also investigate a multi-MTSO scenario given that this new module will allow the AETOS Primary controller to talk to multiple MTSOs.

5.2.3 Prototyping Controller Redundancy High Availability scenarios

For an AETOS deployment to be carrier-grade, high availability must not be an after thought but rather a key consideration of the design. To this end, a future work may investigate EQUAL mode or MASTER-SLAVE configuration modes for Openflow controllers and benchmarks scenarios such as controller failures and switchovers.

5.2.4 Extending AETOS to TCP based flows

The focus in this thesis has been exclusively on UDP based flows mainly in part because the applications investigated in this thesis have been UDP based such as online gaming and video streaming. However a major portion of applications on the mobile network are TCP based. A future work item may investigate expanding AETOS to support agnostic

offloading over a TCP based transport. This will be a bigger challenge given that a TCP sessions is negotiated end-to-end between the P-GW packet gateway and the UE. Packets injected into the session from the offloading gateway in the downlink direction run the risk of TCP checksum failures at the UE. Support for TCP session offloading in AETOS will facilitate a wider adoption of this framework within the mobile community.

5.2.5 OSS/BSS interactions scenario with AETOS Orchestration Agent

The scope of this work has been restricted to the internals of the AETOS framework. A future work may explore the interactions of an external OSS/BSS system to the AETOS Orchestration agent and the different business logic scenarios that would trigger offloading. Such an effort would go a long way towards an end-to-end deployment of such a framework.

5.2.6 OF Controller Service Function Chaining scenarios for dynamically installed GTP parsing middleboxes

While we did briefly touch on dynamic allocation of the middlebox, within our reference implementation of AETOS, the GTP middlebox is statically provisioned which has the disadvantage of permanently reserving MNO resources even during low traffic periods where offloading may not necessarily be required. Most modern controllers allow dynamic Service Function Chaining (SFC). OpenDaylight (ODL) [4] is one such Controller platform which prides itself in being carrier-grade and is widely adopted by a number of industry partners. ODL provides an SFC implementation [3] which may be used to dynamically install the GTP parsing middlebox as driven by a trigger from the Orchestration Agent. A future work item may explore this idea and attempt to deploy ODL service chains.

References

[1] Detailing the difference between simulators and emulators in software development. http://newscenter.ti.com/index.php?s=32851&item=126331. Accessed: 2016-01-30.

[2] NS3: Queues. https://www.nsnam.org/docs/models/html/queue.html.

[3] ODL: Service Function Chaining Reference Use Case. https://www.opendaylight.org/UseCaseSFCReference.

[4] Opendaylight platform. https://www.opendaylight.org/. Accessed: 2016-01-30.

[5] The openflow 1.3 module for ns-3. http://www.lrc.ic.unicamp.br/ofswitch13/index.html. Accessed: 2016-01-30.

[6] Openflow switch support. https://www.nsnam.org/docs/release/3.13/models/html/openflow-switch.html.

[7] Project floodlight. http://www.projectfloodlight.org/floodlight/. Accessed: 2016-01-30.

[8] Software Defined Networking: The New Norm for the Networks. Open Networking Foundation White Paper, April 2012.

[9] Global Games Market Grows to $86.1bn in 2016. http://www.newzoo.com/press-releases/global-games-market-grows-to-86-1bn-in-2016, October 2013.

[10] Traffic Simulation across IP, Analog, TDM, and Wireless Networks with MAPS. http://www.gl.com/traffic-simulation.html, 2014.

[11] Mojdeh Amani, Toktam Mahmoodi, Mallikarjun Tatipamula, and Hamid Aghvami. Sdn based data offloading for sdn based data offloading for 5g mobile networks g mobile networks. *ZTECOMMUNICATIONS*, page 34, 2014.

[12] Nicola Baldo. The ns-3 lte module by the lena project.

[13] Nicola Baldo, Marco Miozzo, Manuel Requena-Esteso, and Jaume Nin-Guerrero. An open source product-oriented lte network simulator based on ns-3. In *Proceedings of the 14th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, pages 293–298. ACM, 2011.

[14] Arijit Banerjee, Xu Chen, Jeffrey Erman, Vijay Gopalakrishnan, Seungjoon Lee, and Jacobus Van Der Merwe. Moca: a lightweight mobile cloud offloading architecture. In *Proceedings of the eighth ACM international workshop on Mobility in the evolving internet architecture*, pages 11–16. ACM, 2013.

[15] Arsany Basta, Wolfgang Kellerer, Marco Hoffmann, Karel Hoffmann, and E-D Schmidt. A virtual sdn-enabled lte epc architecture: a case study for s-/p-gateways functions. In *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*, pages 1–7. IEEE, 2013.

[16] Siwar Ben Hadj Said, Malla Reddy Sama, Karine Guillouard, Lucian Suciu, Gael Simon, Xavier Lagrange, and Jean-Marie Bonnin. New control plane in 3gpp lte/epc architecture for on-demand connectivity service. In *Cloud Networking (CloudNet), 2013 IEEE 2nd International Conference on*, pages 205–209. IEEE, 2013.

[17] Gustavo Carneiro. Ns-3: Network simulator 3. In *UTM Lab Meeting April*, volume 20, 2010.

[18] Gustavo Carneiro, Pedro Fortuna, and Manuel Ricardo. Flowmonitor: a network monitoring framework for the network simulator 3 (ns-3). In *Proceedings of the Fourth*

International ICST Conference on Performance Evaluation Methodologies and Tools, page 1. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009.

[19] KJS Chadha, CF Hunnicutt, SR Peck, and J Tebes. Advanced mobile phone service: Mobile telephone switching office. *Bell System Technical Journal*, 58(1):71–95, 1979.

[20] Roman Chertov, Sonia Fahmy, and Ness B Shroff. Fidelity of network simulation and emulation: A case study of tcp-targeted denial of service attacks. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 19(1):4, 2008.

[21] Margaret Chiosi, Don Clarke, P Willis, A Reid, J Feger, M Bugenhagen, W Khan, M Fargano, C Cui, H Deng, et al. Network functions virtualisation introductory white paper. In *SDN and OpenFlow World Congress*, 2012.

[22] Junguk Cho, Binh Nguyen, Arijit Banerjee, Robert Ricci, Jacobus Van der Merwe, and Kirk Webb. Smore: software-defined networking mobile offloading architecture. In *Proceedings of the 4th workshop on All things cellular: operations, applications, & challenges*, pages 21–26. ACM, 2014.

[23] Cisco Visual Networking Index Cisco. Global mobile data traffic forecast update, 2013–2018. *white paper*, 2014.

[24] OpenFlow Switch Consortium et al. Openflow switch specification version 1.1. 0 implemented (wire protocol 0x02), february 2011.

[25] M Corici, F Gouveia, T Magedanz, and D Vingarzan. Openepc: a technical infrastructure for early prototyping of next generation mobile network testbeds. *Tridentcom, Berlin*, 2010.

[26] Christopher Cox. *An introduction to LTE: LTE, LTE-advanced, SAE and 4G mobile communications*. John Wiley & Sons, 2012.

[27] Souheir Eido and Annie Gravey. How much lte traffic can be offloaded? In *Advances in Communication Networking*, pages 48–58. Springer, 2014.

[28] Dino Farinacci, Darrel Lewis, David Meyer, and Vince Fuller. The locator/id separation protocol (lisp). 2013.

[29] Harald T Friis. A note on a simple transmission formula. *Proceedings of the IRE*, 34(5):254–256, 1946.

[30] Raj Jain, Dah-Ming Chiu, and William Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. 1998.

[31] Xin Jin, Li Erran Li, Laurent Vanbever, and Jennifer Rexford. Softcell: Scalable and flexible cellular core network architecture. In *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*, pages 163–174. ACM, 2013.

[32] Takeshi Kitahara, Antti Riikonen, and Heikki Hämmäinen. Characterizing traffic generated with laptop computers and mobile handsets in gprs/umts core networks. In *Local Computer Networks (LCN), 2010 IEEE 35th Conference on*, pages 1046–1053. IEEE, 2010.

[33] Haneul Ko, Giwon Lee, Insun Jang, and Sangheon Pack. Optimal middlebox function placement in virtualized evolved packet core systems. In *Network Operations and Management Symposium (APNOMS), 2015 17th Asia-Pacific*, pages 511–514. IEEE, 2015.

[34] Amit Kumar, Jyotsna Sengupta, and Yun-fei Liu. 3gpp lte: The future of mobile broadband. *Wireless Personal Communications*, 62(3):671–686, 2012.

[35] Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, page 19. ACM, 2010.

[36] Alcatel Lucent. The lte network architecturea comprehensive tutorial. *Strategic Whitepaper*, 2009.

[37] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation

in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008.

[38] Kanthi Nagaraj and Sachin Katti. Procel: Smart traffic handling for a scalable software epc. In *Proceedings of the third workshop on Hot topics in software defined networking*, pages 43–48. ACM, 2014.

[39] CISCO Global Visual Networking and Cloud Index. Forecast and methodology, 2011-2016.

[40] Mike Owens and Grant Allen. *SQLite*. Springer, 2010.

[41] George F. Riley and Thomas R. Henderson. *Modeling and Tools for Network Simulation*, chapter The ns-3 Network Simulator, pages 15–34. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[42] CB Sankaran. Data offloading techniques in 3gpp rel-10 networks: A tutorial. *Communications Magazine, IEEE*, 50(6):46–53, 2012.

[43] OpenFlow Switch Specification-Version. 1.4. 0, 2013.

[44] Tarik Taleb. Toward carrier cloud: Potential, challenges, and solutions. *Wireless Communications, IEEE*, 21(3):80–91, 2014.