DEGREE PROGRAMME IN WIRELESS COMMUNICATIONS ENGINEERING

MASTER'S THESIS

Wireless Backhaul in Future Cellular Communication

Author	Munim Morshed
Supervisor	Mika Ylianttila
Second Examiner	Jari Iinatti
(Technical Advisor	Jaakko Leinonen)

August 2018

Morshed Munim. (2018) Wireless Backhaul for Future Cellular Communication. University of Oulu, Degree Programme in Wireless Communications Engineering. Master's Thesis, 64 p.

ABSTRACT

In 5G technology, huge number of connected devices are needed to be considered where the expected throughput is also very ambitious. Capacity is needed and thus used frequencies are expected to get higher (above 6 GHz even up to 80 GHz), the Cell size getting smaller and number of cells arising significantly. Therefore, it is expected that wireless backhaul will be one option for Network operators to deliver capacity and coverage for high subscriber density areas with reduced cost. Wireless backhaul optimization, performance and scalability will be on the critical path on such cellular system. This master's thesis work includes connecting a base station by using the wireless backhaul by introducing a VPN in the proposed network. We find the bottleneck and its solution. The network is using 3.5 GHz wireless link instead of LAN wire for backhaul link between the EnodeB and the core network (OpenEPC). LTE TDD band 42 acting as a Wireless Backhaul (Link between EnodeB and Band 42 CPE Router). The status and attachment procedure are observed from different nodes of the openEPC and from the VPN machine. Step by step we have established a tunnel between the CPE device and the VPN server using PPTP and L2TP with IPSec tunneling protocol. The progression towards the final implementation brings in step by step all difficulties and bottlenecks are documented in the study.

Key words: 5GTN, LTE, CPE, IPSec, L2TP, PPTP, Throughput, Latency.

Table of Contents

ABSTRACT 2
FOREWORD 5
LIST OF ABBREVIATIONS AND SYMBOLS 6
1. INTRODUCTION 9
1.1. 5G Test Network, University of Oulu 9
1.2. Aims and Objective 11
1.3. Thesis Structure 12
2. WIRELESS BACKHAUL 13
2.1. Existing Wireless Backhaul Technology 13
2.2. Requirements of wireless backhaul in cellular network 14
2.3. Bottlenecks of implementation 15
3. VPN TUNNELLING PROTOCOLS OF INTEREST 17
3.1. VPN tunnelling Protocols 17
3.2. Layer 2 Tunnelling Protocol (L2TP) 17
3.3. L2TP Protocol Stack 18
3.4. IPSec Tunneling 19
3.4.1. Authentication Header 20
3.4.2. IPSec Encapsulation Security Payload (ESP) 20
3.4.3. IPSec Protocol stack 21
3.4.4. IPSec Key Exchange 21
3.5. Point to Point Tunneling Protocol (PPTP) 22
3.6. PPTP Protocol Stack 23
4. IMPLEMENTATION AND MEASUREMENT SETUP 24
4.1. Customer Premises Equipment (CPE) 25
4.2. SoftEther VPN Server 26
4.3. Tunneling between the server and the CPE device 28
4.3.1. layer2tp.softether.net server settings and configuration 29
4.3.2. CPE1 and CPE 2 Configuration 30
4.4. Tunneling between the PC(client) and the server 32
4.5. Tunneling between the CPE device and VPN software 36
4.6. Openswan VPN Server 39
4.7. Tunneling between PC (client) and L2TP/IPSec Openswan VPN 40
4.8. Tunnelling between PC (client) and PPTP Openswan VPN 43
4.9. Tunnelling between CPE (client) and L2TP/IPSec Openswan VPN 45
4.10. Tunnelling between CPE (client) and PPTP Openswan VPN 47
4.11. StrongSwan VPN Server 49
4.12. Tunnelling between CPE(Client) and L2TP VPN Server 50
5. DISCUSSION 56
6. SUMMARY 59
7. REFERENCES 60

8. APPENDICES 62

FOREWORD

The thesis work is supervised and completed at CWC, University of Oulu, 5GTN project as a part of the master's thesis requirement for Wireless Communication Engineering. The thesis work is an effort to have a contribution in the world of research, even though it is small as a drop of water in the ocean. This work is dedicated to those esteemed hard workers who makes differences and bring happiness to life.

"I'm writing a first draft and reminding myself that I'm simply shovelling sand into a box so that later I can build castles." -Shanon Hale

In the whole journey towards the completion of thesis, I have learned a lot of practical things as well as corelating with the theoretical knowledge. I have enjoyed immensely and feel lucky to be a part of the 5GTN project. I would like to thank my technical supervisor, Jaakko Leinonen whose guidance always helped me finding the right path. I would like to thank my supervisor, Professor Mika Ylianttila for his all-out guidance and suggestions at each stage of the work. The whole 5GTN project team members helped me a lot. I would like to mention here, Miika Räisänen, Olli Liinamaa and Muhammad Arif for their unstinted cooperation and help whenever needed.

Oulu, February 12, 2018
Munim Morshed

LIST OF ABBREVIATIONS AND SYMBOLS

3GPP	Third-Generation Partnership Project
5G	5th Generation
5GTNF	5G Test Network Finland
5GvLAN	5G Virtual Local Area Network
AAA	Authentication Authorization and Accounting
ACK	Acknowledgement
AH	Authentication Header
APN	Access Point Name
ARP	Address Resolution Protocol
CHAP	Challenge Handshake Authentication Protocol
CHAP	Challenge Handshake Authentication Protocol
CND	Core Network Dynamics
CPE	Customer Premises Equipment
CPU	Central Processing Unit
CWC	Centre for Wireless Communications
DHCP	Dynamic Host Configuration Protocol
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Server
DSL	Digital Subscriber Line
E2E	End to End
EAP	Extensible Authentication Protocol
EnodeB	Evolved Node B
EPC	Evolved Packet Core
EPC_Enabler	Evolved Packet Core Enabler
ePDG	Evolved Packet Data Gateway
ESP	Encapsulating Security Payload
eth	Ethernet
EUTRAN	Evolve Universal Terrestrial Radio Access Network
FDD	Frequency Division Duplexing
GRE	Generic Routing Encapsulation
GSM	Global System for Mobile Communication
GUI	Graphical User Interface
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
HTTP	Hypertext Transfer Protocol
Hz	Hertz
I/O	Input/output
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
IKE	Internet Key Exchange

IKE-v2	Internet Key Exchange version 2
IMSI	International Mobile Subscriber Identity
IOT	Internet of Things
IP	Internet Protocol
IPSec	Internet Protocol Security
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISAKMP	Internet Security Association and Key Management Protocol
ISP	Internet Service Provider
ITU	International Telecommunication Union
L2TP	Layer 2 Tunnelling Protocol
LAC	L2TP Access Concentrator
LAN	Local Area Network
LNS	L2TP Network Server
LOS	Line of Sight
LS-MIMO	Large Scale Multi Input Multi Output
LTE	Long Term Evolution
MAC	Media Access Control
MCC	Mobile Country Code
MME	Mobility Management Entity
MNC	Mobile Network Code
MPPE	Microsoft point-to-point Encryption
MSIN	Mobile subscriber identification Number
MTU	Maximum Transmission Unit
NAS	Non-Access Stratum
NAT	Network Address Translation
NIC	Network Interface Control
NMSI	National Mobile Subscriber Identity
OpenEPC	Open Evolved Packet Core
OS	Operating System
OSI	Open Systems Interconnection Model
PAC	PPTP Access Concentrator
PAP	Password Authentication Protocol
PAP	Password Authentication Protocol
PC	Personal Computer
PDN GW	Packet Data Network Gateway
PGW	Packet Gateway
PLMN	Public Land Mobile Network
PNS	PPTP Network Server
PoC	Proof of Concept
PPP	Point to Point Protocol
PPP	Point to Point Protocol
PPTP	Point to Point Tunnelling Protocol

QoS	Quality of Service
RAN	Radio Access Network
RAS	Remote Access Server
RFC	Request for Comments
RFC	Request for Comments
RNC	Radio Network Controller
S1AP	S1 Application Protocol
SA	Security Association
SCTP	Stream Control Transmission Protocol
SCTP	Stream Control Transmission Protocol
SGW	Serving Gateway
SIM	Subscriber Identity Module
SPGW	Serving Packet Gateway
SRN	Shared Resource Network
SNAT	Source Network Address Transilation
SSL	Secure Socket Layer
SSL	Secure Sockets Layer
SSTP	Secure Socket Tunnelling Protocol
TCP	Transmission Control Protocol
TDD	Time Division Duplexing
TLS	Transport Layer Security
UDP	User Datagram Protocol
UE	User Equipment
UE	User Equipment
UMTS	Universal Mobile Telecommunication Systems
UTRAN	Universal Terrestrial Radio Access Network
vLAN	Virtual Local Area Network
VM	Virtual Machine
VPN	Virtual Private Network
VPN	Virtual Private Network
WAN	Wide Area Network
WEP	Wired Equivalent Privacy
WPA	Wi-fi Protected Access

1. INTRODUCTION

Cellular mobile and data communication system are expanding rapidly for the demand of more coverage and capacity. The use of high speed internet is a requirement for many applications such as when online gaming requires high data rate and lower latency to achieve real time experience. This growing need of capacity can be fulfilled using higher frequency bands; inevitably the cell size would be significantly smaller and number of cells will arise consequently. The communicating base stations are linked to one to another and to the core network using wireless backhaul. Thus, wireless backhaul is one of the major concern for future 5G cellular communication. In this thesis we have proposed a design/topology to provide improved data rate and capacity for distant users. In the network the pico and Macro base stations are connected to virtual core 'OpenEPC[1]' at the premises of University of Oulu. We have performed tests at different points over the network for throughput and latency measurement. The thesis will go through the details step by step.

1.1. 5G Test Network, University of Oulu

5GTN is a joint effort from industry, academia and Finnish Govt. with globally recognized companies such as Nokia, Huawei, Ericsson, Coriant and intel and also internationally recognized research organizations like VTT, University of Helsinki, University of Oulu, Tampere University of Technology and Aalto University. 5GTN is a part of the project 5GTNF which is funded by Tekes-the Finnish funding agency for innovation aimed to research, business development and innovation purposes. It is the joint collaboration of the research teams both academic and industry to facilitate national and international collaboration in research and development for future 5G technology.

5GTN[2] have already built a fully functional LTE network with all necessary infrastructure to use it as a 5G proof of concept testbed for future 5G cellular network.

5GTN has a complete network infrastructure where some network functions are shared, maintained and managed for the fully functional LTE network, which is a step ahead towards 5G. The 5GTN testbed is also connected with Oulu University hospital which is a few kilometers away from university premises for medical IOT solutions and used cases. A more detailed information about the 5GTN testbed is shown in figure 1:

[1] OpenEPC by CND; Details available at http://www.openepc.com/home/overview/

[2] 5G Test Network; Available at http://5gtn.fi/

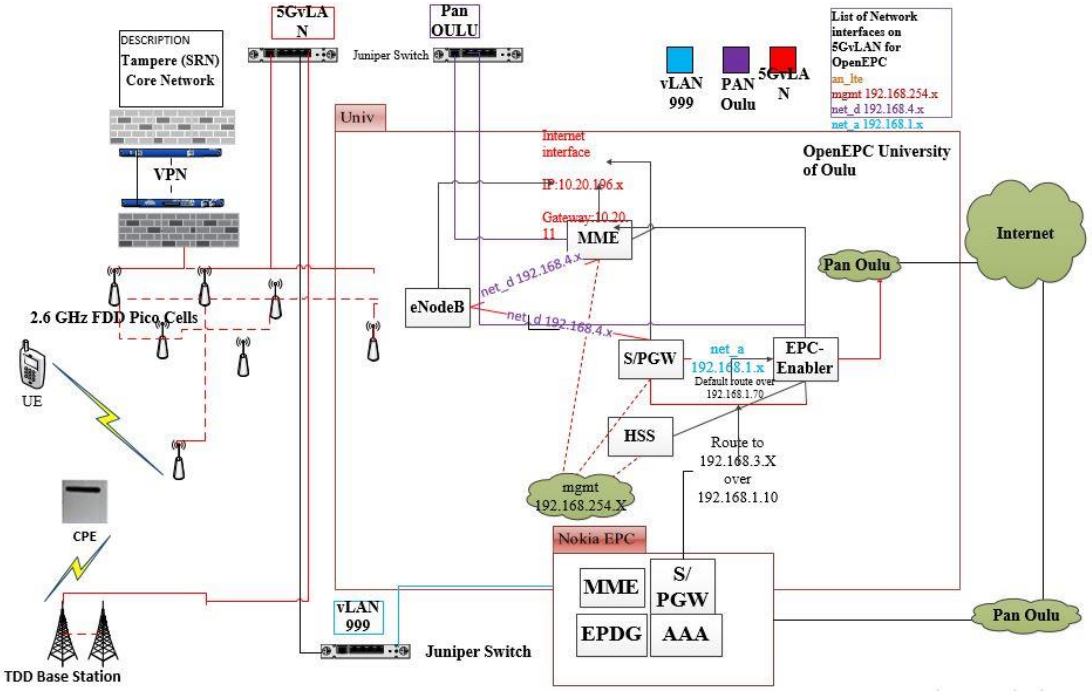

Figure 1. 5G Test Network Architecture [1].

5GTN is currently using 2.6GHz (LTE Band 7) for the FDD pico cells and 3.5GHz (LTE Band 42) for the TDD Macro base station. The band 3.5 GHz (3.4-3.6 GHz) is supposed to be used only in the university premises for research purposes as it is commercially unlicensed. A number of Pico and Macro cells are installed and commissioned in the university premises, allocating the configuration of the MME according to the control and data plane of respective core network. Thus a mobile/end device knows to which core network it needs to connect. There are three different core networks which is shown in figure 1. OpenEPC is a virtual core network deployed at University of Oulu IT servers, where it is running on VMWare ESXi vSphere environment. The nodes of OpenEPC such as MME, HSS, EnodeB, SPGW and EPC-enablers are installed and running in virtual machines. A VPN server is installed in another virtual machine 'vpn5g' along with other VM's. The VPN server is used to establish an IP security tunneling between the server and the CPE device.

The second core network is a shared resource network (SRN) core, situated in Tampere, Finland owned by Nokia. It is connected to the 5GTN network via VPN over internet. Lastly we have Nokia EPC at the vLAN 999 at the university premises, which is on the way to installation. OpenEPC is a virtual LTE network designed by Core Network Dynamics (CND). Users are provisioned with unique IMSI and CND test SIM card. It is deployed in the cloud environment, in the servers of university of Oulu and can be accessed remotely from the network of university of Oulu i.e 'Univ domain'. 5GTN can be used as a testbed for the integration of devices and built infrastructure for both 3gpp and non 3gpp cases. The testbed has successfully integrated base stations and gateways to access and collect data from different IOT sensors[3] [1].

[3] 5G Test Network project page available at http://5gtn.fi

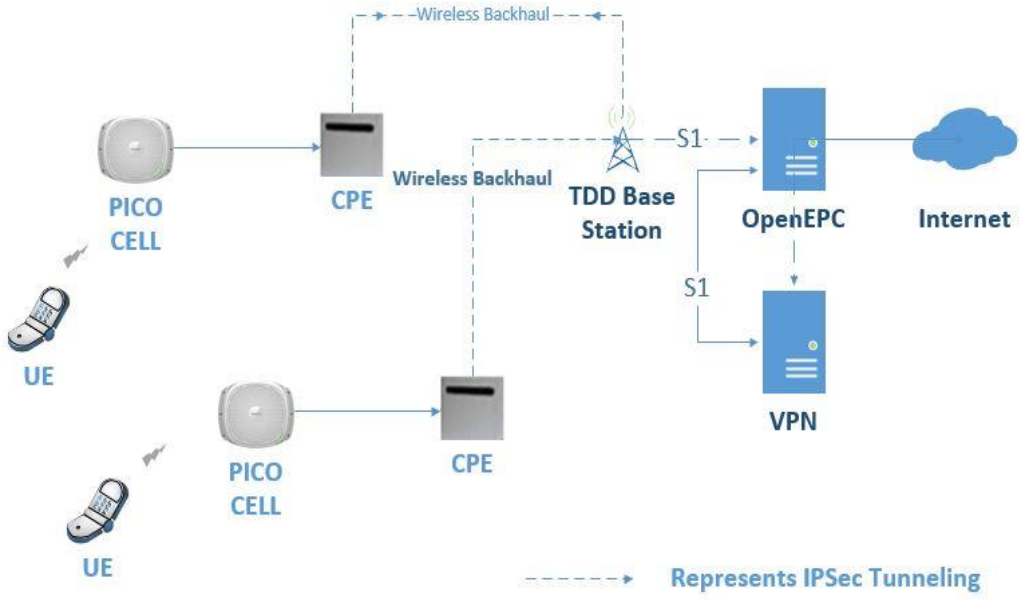

Figure 2. Pico Base stations connected via CPE device

1.2. Aims and Objective

The scope and motivation for this thesis is how wireless backhaul implementation and observation of the herein used case. A closed observation on the performance evaluation based on throughput and delay analysis is implemented in the network. An important outcome of the work demonstrates the easy and hustle free plug and play installation of the small cells. These small cell implementation enables mobility. For example, if we need to give coverage and capacity to some low coverage place, the small cells (Pico base station) can be easily carried with the CPE device to the desired location and can be connected with the core network. Thus, we can achieve high coverage and capacity in the network. Implementation of small cells and trails are demonstrably performed to see the outcome performance of the small cells. In the above figure multiple base stations can be connected via CPE device. CPE device connects to the nearest TDD base station which is connected to the core network. In the configuration of pico cells, the specific IP address of the HSS of specific core network is provided so that the users can be authenticated getting access from that specific core network. In this case the S1 connectivity established via Wireless backhaul. In the Pico base station configuration, we have changed the MME IP to the IP address of the OpenEPC MME changing the PLMN accordingly. Thus, any user using the OpenEPC SIM card can be authenticated from the core network and get connected.

1.3. Thesis Structure

In thesis writing we have presented the related theories with the corresponding works compiled in a step by step approach to make it more understandable for the new readers. In Chapter 1 a brief overview is narrated to understand the present network scenario of the testbed. Chapter 2 describes the basic information of wireless backhaul in cellular systems. Chapter 3 describes the tunneling protocols and their working mechanism including the related theories. We have discussed the protocols of our interest. Chapter 4 describes the implementation of security tunneling in different platform archived step by step. We have implemented by starting with some test configurations and gradually step by step implemented our required setups on every step, we have documented all related literatures. The thesis is written as a compiled journal describing setup architecture and bottlenecks for the better understanding for the new readers that be.

2. WIRELESS BACKHAUL

Wireless backhaul is the wireless communication or network infrastructure responsible for transporting data/traffic from end user to the central network infrastructure and vice versa. The backhaul is expected to play a critical role in handling huge data traffic, capability driven by the growing demand from both mobile broadband and deployment of heterogeneous networks. (HetNets) [2]. The simplified network architecture can be split into Air interface, Radio Access Network and core network. The link between the Radio Access network and the Core network is referred to as backhaul. Generally, for GSM networks these backhaul links are E1and T1 links which are used to carry low data rates. The E1 and T1 can carry 2Mbps and 1.5 Mbps respectively.

2.1. Existing Wireless Backhaul Technology

From 3G to beyond towards 5G it is an essential requirement to have high data rate support from the backhaul. There are many different backhaul types and sub categories that are needless to mention here. Only the major backhaul types are listed below:

- Wired
  - Optical (Fiber)
  - (A)DSL
- Wireless
  - Microwave (10-42 GHz)
  - V Band/60 GHz (57-64 GHz)
  - E-Band (71-76, 81-86 GHz)
- Satellite
- Wi-Fi Mesh
- LTE Backhaul
- TD-LTE Backhaul
- Self-backhauling
- Optical
  -Laser

In case of wired backhaul, the most popular type is fiber backhaul which is mainly used in 4G networks. For more high data rate Optical backhauling is needed. Coaxial cables and Ethernet cables are used for backhauling in 3G networks. Speaking of microwave frequencies microwave plays a vital role in wireless backhaul. It is hard to deploy cable connection in difficult terrain and environment conditions. Microwave frequencies are preferable and practical so there is traditional 10 to 42 GHz microwave link. Then there is V band which is 57 to 64 GHz frequency, which is generally referred to as 60 GHz band. There is also the E band 71 to 76 GHz and 81 to 86 GHz band. These ITU band names are different from the 3GPP band names. In future 5G communication network multiple layers of frequencies are needed to provide a good and efficient network. For example, 1 to 6 GHz can be used to provide capacity and for higher throughput 6 to 100 GHz can be utilized. Satellite backhauls are especially used for rural and remote locations. Traditional backhaul

technology cannot reach such areas thus satellite backhaul plays the vital role in communicating remote locations. LTE backhaul can be used in “in Band” or “Out of Band” approach. TD LTE backhaul can be deployed for improved coverage and capacity. In 5G it is expected that there would be dense deployment so only fiber cannot connect each and every base station. For example, there might be some situation when one base station connected to fiber and thereby it connects three or four base stations by self-backhauling. Optical backhauling can be an expensive solution which uses point to point connectivity and LOS which has the capacity of high throughput[4].

2.2. Requirements of wireless backhaul in cellular network

The backhaul technology is the infrastructure that is responsible to connect small networks to the backbone or the primary network. Small cells/ Pico cells can be connected to the donor base station i.e. Macro Cell through wireless backhaul as illustrated as in the figure below:

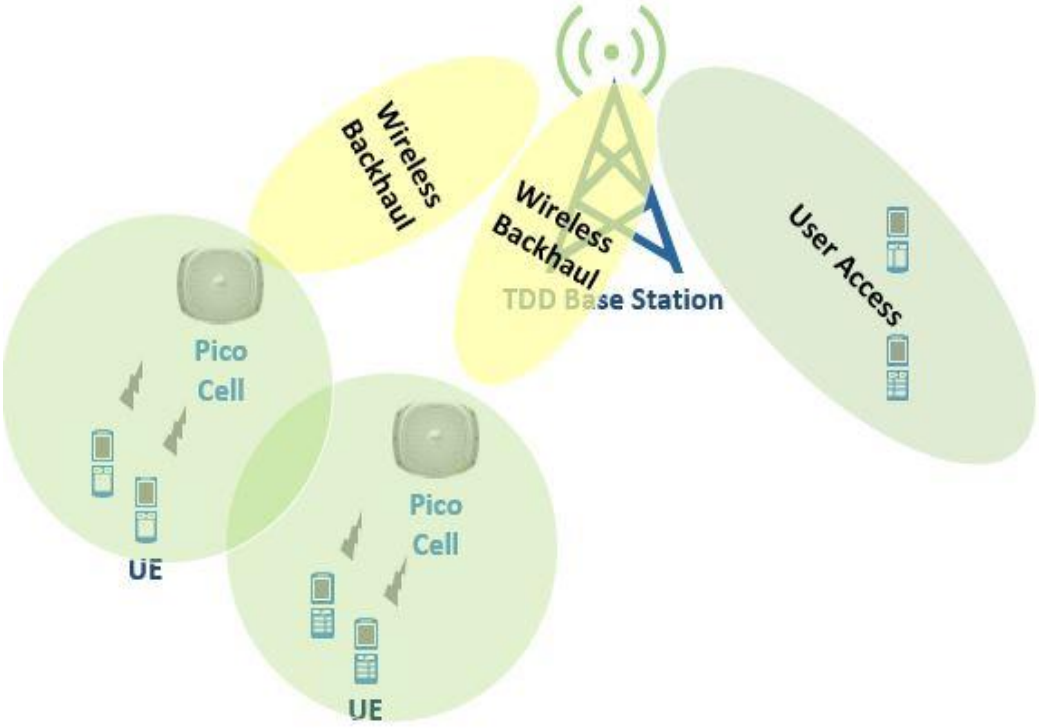

Figure 3. Wireless Backhaul in heterogeneous network.

The wired solution may provide high reliability, high capacity, high speed but wireless backhaul is preferable as it offers easy deployment and economical solution. However, to choose a wireless backhaul solution in a heterogeneous network depends on some system considerations, such as network capacity, small cell density, electromagnetic interference, data rate requirement etc, which must be ensured

[4] 3G 4G Wireless Resource Centre; Available at http://3g4g.co.uk/

maintaining a low power consumption and availability of orthogonal backhaul spectrum. Thus, in a heterogeneous network having both Micro cells and small cells forward huge traffic to the core network can be a critical issue. Wireless backhaul demands high throughput and low latency, which is the challenge of future 5G cellular communication. LS-MIMO is going to be deployed for which high performance interference mitigation capacity is required. The small cells are deployed and increasing in density for which Energy Efficient technique such as beamforming can be deployed to reduce the power consumption of the wireless backhaul [3].

2.3. Bottlenecks of implementation

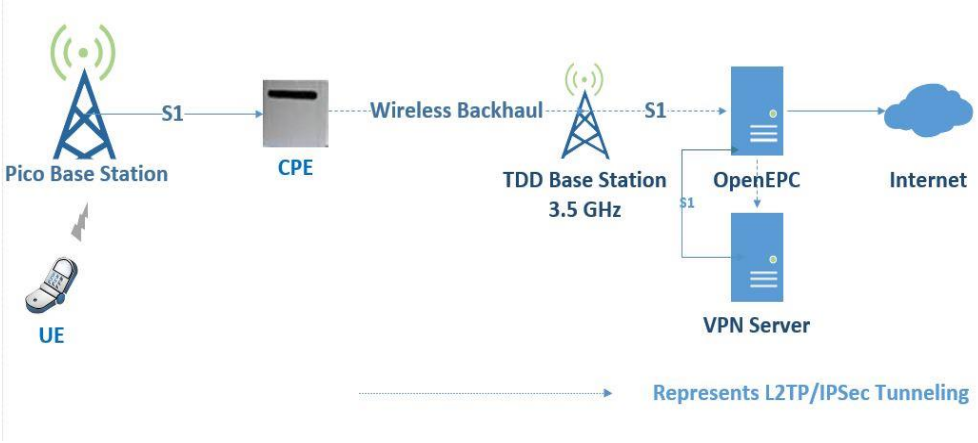

Figure 4. Implementation Setup

In simplest form our implementation is presented in the above figure. In the network the CPE device is placed amidst two base stations where TDD base station already has S1 connection with the OpenEPC. To connect pico base station, it needs to establish a S1 connection with the MME. It is one of the hindrance of this arrangement because we cannot pass one S1 through another S1. To overcome this problem, we have introduced a VPN server, which is connected to the OpenEPC. The CPE device running in tunneling mode enables the pico base station to make S1 connection to the core network. The CPE device and the TDD base station both operating at 3.5 GHz, have wireless backhaul connection between them. In the above figure the dotted line represents L2TP/IPSec tunneling. The tunneling starts at the client CPE device and ends at the end of the tunnel i.e server. The VPN server is running in a virtual machine “vpn5g”. In our system we have used Openswan L2TP/IPSec VPN server to serve our purpose. The VPN server is connected to PanOulu. On the other hand, the nodes of OpenEPC are also connected to PanOulu like figure 4. The CPE device and the user mobile phone must use OpenEPC SIM card to have authentication and get connected. Different VPN types with different tunneling protocols can be implemented. The CPE device we are using have different tunneling protocols with L2TP and PPTP protocols. In this thesis our discussion will be limited about discussing these two types only because the VPN server we have used here have the capability to support these two protocols only. The other protocol types are hard to be compatible with being obsolete otherwise.

In our system, primarily we implement OpenSwan L2TP/IPSec VPN server. The operating system of the VPN machine was ubuntu. OpenSwan appear to be not compatible with ubuntu 16.04 but runs in ubuntu 14.04 and Debian OS. On the other hand, PPTP tunnelling protocol is obsolete over time and in most of the Linux OS do not support properly for which the use of PPTP in practical applications become limited.

In our final implementation, which is a stand-alone setup where the software VPN (StrongSwan) is running on Debian OS, we observed this time that the IPSec is not working and the PPTP has similar dis-compatibility issues. The L2TP tunnel works perfectly by which we establish the S1 connectivity.

3. VPN TUNNELLING PROTOCOLS OF INTEREST

3.1. VPN tunnelling Protocols

Tunneling protocol allows a network user access and provide network service which is done by encapsulating a packet from one type of protocol within the datagram of a different protocol. For example, PPTP tunneling protocol can be used to encapsulate IP packets over the internet. Depending on the requirements and capabilities of network devices, we can choose the tunneling protocols. There are many tunneling protocols such as Layer 2 Tunneling Protocol (L2TP), Point to Point Tunneling Protocol (PPTP), Generic Routing Encapsulation (GRE), Secured Socket Tunneling Protocol (SSTP) etc. Whereas IPSec protocol is cryptographic security services to protect communication over Internet Protocol (IP) to confidentiality [4].

3.2. Layer 2 Tunnelling Protocol (L2TP)

L2TP is a standard IETF VPN protocol defined by RFC2661 [5] tunneling point-to-point protocol (PPP) session based on UDP. L2TP extends the nature of PPP. L2TP basically has three components. They are user PC, LAC and LNS. LAC is the L2TP access concentrator. In the PSTN network it accepts PPP requests coming in from the user PC. This is basically the gateway between the user and the LNS or the L2TP network server. LNS is the L2TP network server and its work is to access the terminating point for all the L2TP session coming in from various ways. It sees LNS contains basically a router, a device which acts as the terminating point for L2TP session.

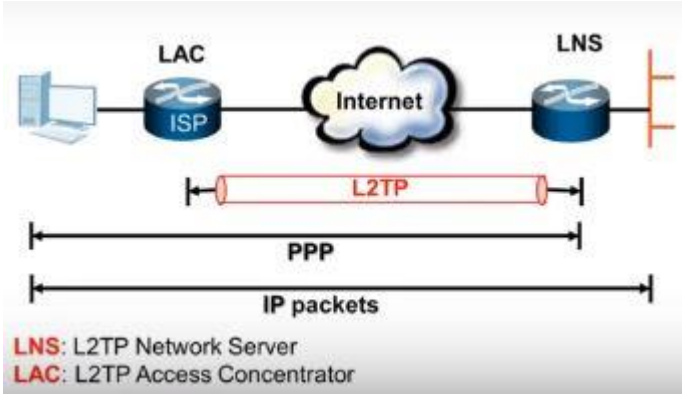

Figure 5. L2TP Structure[5]

User establishes a PPP session to the LAC and LAC will in turn establish an L2TP tunnel between itself and LNS and it will tunnel all the PPP traffic to the LNS, which will then de-capsulate and passes that PPP traffic over the private network. The user is dialing out to the LAC and pass PPP credentials. PPP credentials can pass through PAP, CHAP or MSCHAP authentication scheme. The user locks a PPP session between the user and the LAC sensing its credential using PAP, CHAP or MSCHAP. The LAC authenticate the user and then establishes the L2TP session.

[5] L2TP structure, DLS Reports; available at http://www.dslreports.com/forum/r30248402-ZyXel-USG40-Multiple-IP-Ranges-for-L2TP-VPN

LAC authenticate the user first and then start to setup the L2TP tunnel between itself and LNS. L2TP has two types of messages: control and data messages. The control messages are used for establishing L2TP tunnels and data messages are used by data sessions that encapsulate PPP sessions [6], [7]. L2TP does not encrypt data stream hence not able to provide confidentiality for the network user. Thus, IPSec is used to secure L2TP packets by providing authentication, integrity and confidentiality. A L2TP packet structure is shown in figure 6.

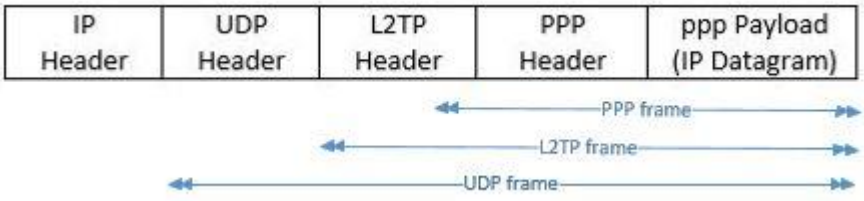

Figure 6. Structure of an L2TP Packet Containing an IP Datagram

In the figure the layers where the data can be encrypted are shown in gray color. Encryption of L2TP traffic with IPSec ESP is shown in figure 7. In First layer of L2TP encapsulation, IP datagram is wrapped with a L2TP header and a UDP header. In second layer encapsulation the L2TP message is wrapped with ESP (IPSec Encapsulation Security Payload) header, a trailer and an IPSec Authentication trailer to provide message integrity and authentication. Figure 7 illustrates L2TP and IPSec encapsulation for a PPP datagram[6].

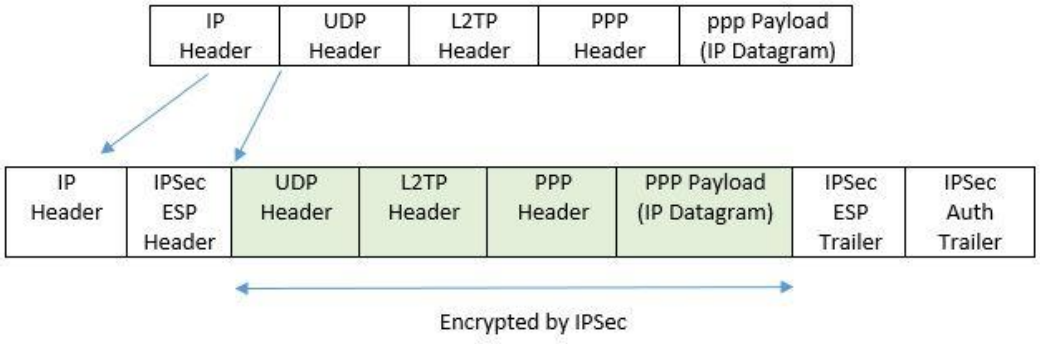

Figure 7. Encryption of L2TP traffic with IPSec ESP

3.3. L2TP Protocol Stack

[6] VPN Tunnelling Protocols; Available at https://technet.microsoft.com/en-us/library/cc771298(v=ws.10).aspx

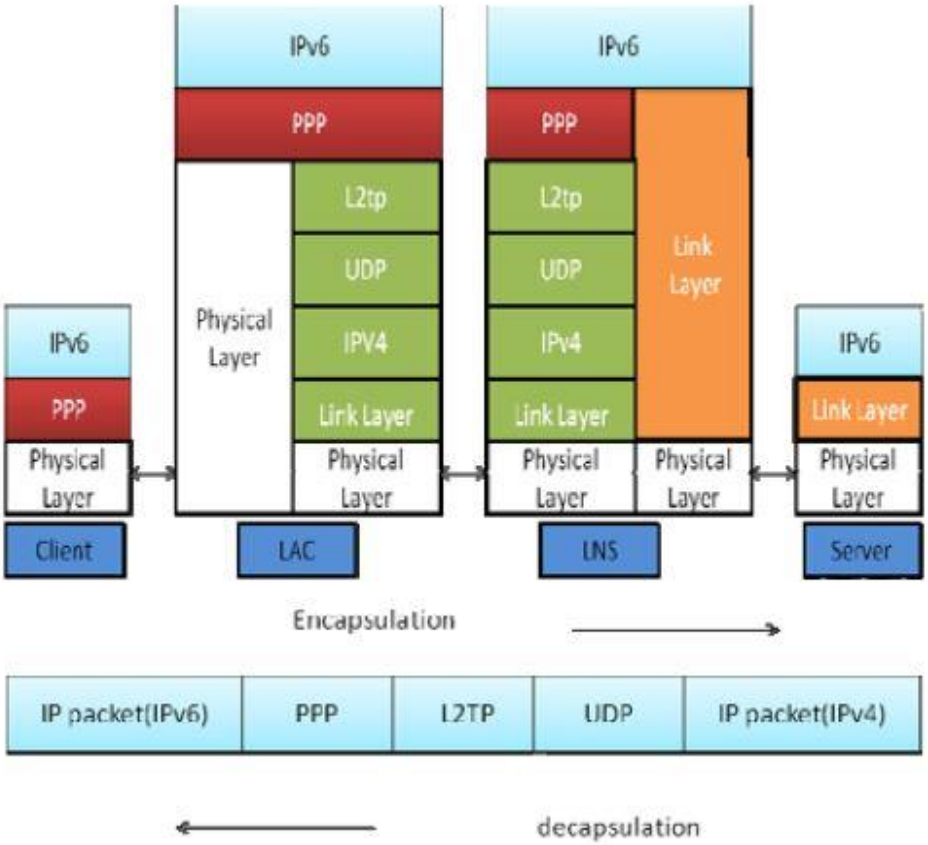

Figure 8. L2TP Protocol Stack [5]

Fig.8 illustrate the encapsulation and stack structure for LAC and LNS. At first, when IPv6 packets forwarded into LAC, it is encapsulated in the PPP interface. Then it is passed as a user data to the L2TP protocol, where it is encapsulated as L2TP payload. Then it is encapsulated into a UDP packet, which is further encapsulated as an IPv4 packet. At this point the source address and destination address is IPv4 address of LAC and LNS respectively. After the tunnel establishment, the network traffic can run in both direction between the peers [7].

3.4. IPSec Tunneling

IPSec is a layer 3 (Network layer) protocol designed to allow data packets to be encrypted and encapsulated for secure transfer across an IP network. IPSec is composed of a set of security services. These services include access control, connectionless integrity, data origin authentication, protection against replays and encryption i.e. confidentiality. There are two traffic security protocols: authentication header (AH) and encapsulating security payload (ESP) plus the cryptographic key management procedures and protocols [8].

3.4.1. Authentication Header

Authentication Header provides data integrity and authentication of IP packets. The authentication procedure is based on a Message Authentication Code (MAC), which is encoded with a keyed hash algorithm, the result of the hash is appended to the packet before it is sent to the recipient. The hash is produced from keying hash algorithm procedure on the data message and the common shared secret, which is shared between the sender and the receiver [9]. AH is supported to two different modes- the transport mode and tunnel mode. In tunnel mode AH appends a new IP header for the whole IP packet. At the recipient end by checking the hash values the message integrity is confirmed, whereas SPI (Security Parameter Index) is an identifier for a successful connection.

At the destination node after receiving the data packet it will regenerate the MAC and will compare with the MAC supplied along with AH header. If it matches then the destination node, send an acknowledgement (ACK) to the sender confirming that authentication and integrity of data has not changed. Otherwise, it will drop the packet and does not send ACK packet. In such case receiving no ACK, the source node will send data again [10].

3.4.2. IPSec Encapsulation Security Payload (ESP)

IPSec Authentication Header (AH) provides integrity, authentication between IPSec being capable of two nodes or devices whereas IPSec Encapsulation Security Payload (ESP) provides integrity and data origin authentication. ESP is used to enable confidentiality by encrypting the payload. There are several services available, depending on the nodes of operation. For example, these services can be of confidentiality only, for integrity only or for both confidentiality and integrity [11].

ESP is supported to operate in tunnel mode or transport mode. In the ESP tunnel mode, a new IP header is appended though the inner IP header stays unchanged whereas in transport mode the ESP header is inserted after the IP header [9]

In case of ESP, in a similar way the destination node will process the ESP header. The intermediate malicious nodes are not able to see the data packet as the encapsulated data packets are encrypted with the shared key and the security association SA is established between the source and destination node. A new IP header which is outside the encapsulated packet is visible to the intermediate malicious node. The original IP header and data packet are encapsulated using the shared key between the source and destination node. The intermediate malicious node cannot change the data packet because of the new IP header being there, which is the only visible thing for it to be seen.

When the packet reached the destination node, it is de-encapsulated using the shared key. After successfully decrypting the packet the destination node sends an ACK to the source to acknowledge that the confidentiality and authenticity has not broken or violated. On the other hand, if the source node doesn't receive any ACK from the destination node, it will transmit the packet again [10].

3.4.3. IPSec Protocol stack

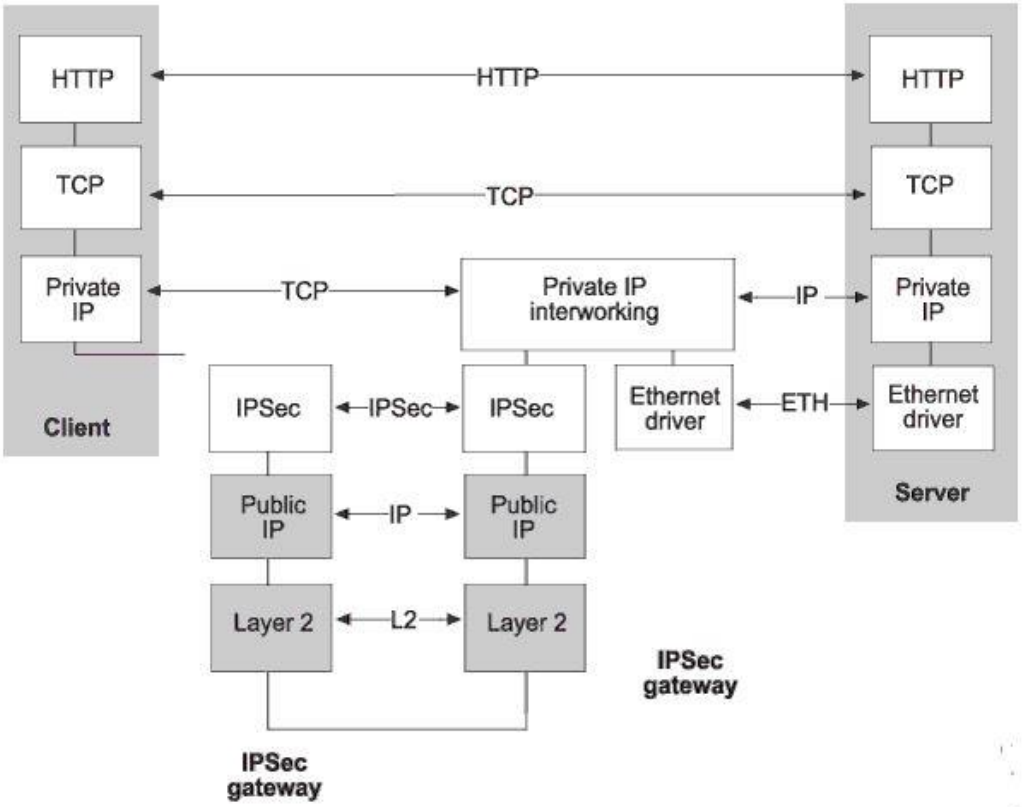

Figure 9. IPSec protocol stack[7]

The figure shows the protocol stack on a client, an IPSec gateway and a server. TCP and HTTP are examples of higher level protocol having end to end (E2E) communication. Other E2E communication protocols are also supported. IPSec works at the Network layer of OSI model and in the IP layer of the conventional TCP/IP protocol stack.

3.4.4. IPSec Key Exchange

The internet Key Exchange (IKE) provides secure communication between two peers using IPSec protocol. Secure communication is established by creating shared security protocols and authentication key between IPSec end points. IKE has two versions: IKEv1 and IKEv2. IKEv1 works in two phases: phase 1 and phase 2. Phase 1 operates in two modes, they are the main mode and the aggressive mode. In main mode, three pairs of messages are used to establish IKE SA. In aggressive mode three messages exchange making it faster but non-reliable. Phase 2 operates in a single mode called quick mode. In quick mode, three messages are exchanged to establish independent IPSec SAs. The messages are unidirectional, and they are

[7]IPSec Concepts; Available at https://www.juniper.net/documentation/en_US/junose10.3/information-products/topic-collections/swconfig-ip-services/id-91638.html

protected by IKE SA. IKEv2 is an extension of IKEv1. In IKEv2 the exchange is initiated by a request message to which the responder generates a response message. If responder does not respond with in a defined time interval, either the request is sent again, or the connection is terminated [9]

3.5. Point to Point Tunneling Protocol (PPTP)

A vendor consortium made up of companies like Ascend Communication, Microsoft, Copper Mountain Networks, 3COM and ECI Telematics developed Point to Point Tunneling Protocol (PPTP) to extend the capabilities and performance of Point to Point Protocol (PPP) which operates at Layer 2 of the OSI model. It creates a secure tunnel between a PPTP Access Concentrator (PAC) and the PPTP Network Server (PNS). In the operation mechanism, first the client uses a PPP connection to build a link between the source and destination through a transit network. Then using Transmission Control Protocol (TCP) session, a connection established between the client and the server. PPTP is used to control the establishment, management and release sessions through the tunnel. Every tunnel will have its own connection control. Finally, the PPP packets containing the original data will be sent through an extended version of Generic Routing Encapsulation (GRE) protocol, encapsulated into IP packets. PPTP does not support IPv6 for which it has limited use [12]. A PPTP frame structure is shown in figure 10.

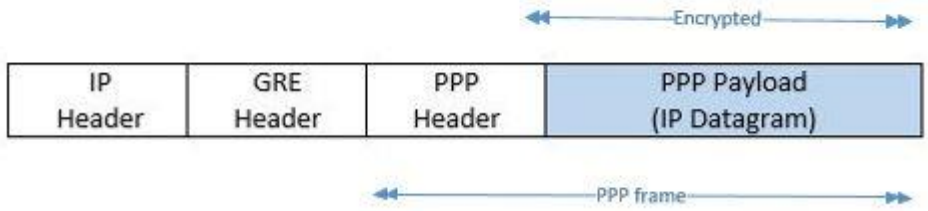

Figure 10. Structure of a PPTP Packet Containing IP Datagram

The PPP frame is encrypted using MPPE (Microsoft Point to Point Encryption) and the encryption keys generated from MS-CHAP v2 or EAP-TLS authentication process.

3.6. PPTP Protocol Stack

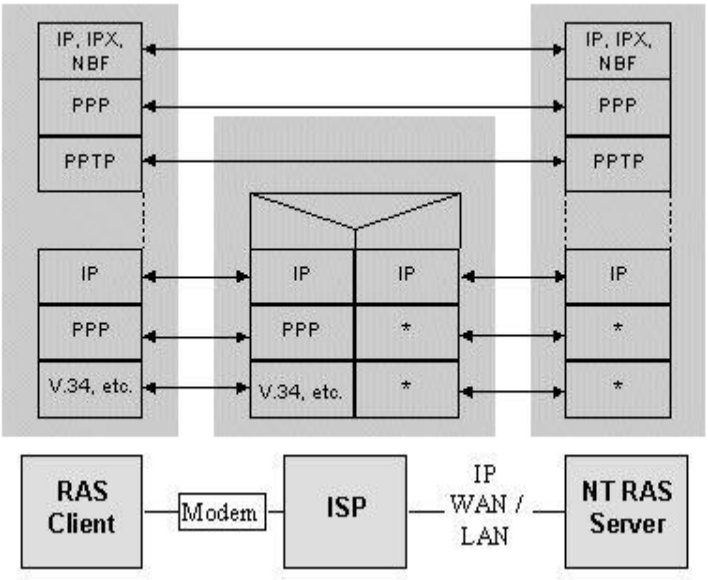

Figure 11. Windows95 PPTP Client/Internet/NT RAS Server Protocol Stack[8]

To run the Windows9x PPTP client establish an IP connection with a tunnel server such as Windows NT Server 4.0 Remote Access Server (RAS). Windows Dial-Up Network uses standard PPP to provide a secure and optimized multiple protocol network connection over dial up telephone line. PPTP encapsulates its data stream in PPP protocol. PPTP is supported to Windows NT, Windows 98, Windows 95 (need update Dial-up Networking 1.2) and many more devices such as modems and CPE devices which are all supported to PPTP protocols.

[8] Configure PPTP server behind SUA; available at:
http://www.zyxeltech.de/SNoteZW5_362/app/pptp.html

4. IMPLEMENTATION AND MEASUREMENT SETUP

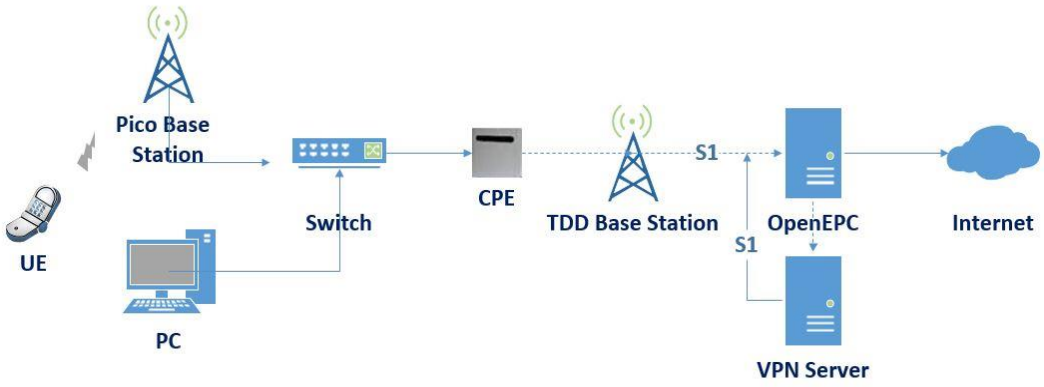

Figure 12. Proposed Network Architecture

In the above scenario, the diagram is implemented to facilitate the remote Pico base station to be connected to the OpenEPC. Here in the figure the dotted line represents the VPN tunneling for which it starts from CPE device and ends at the VPN server. The VPN server is connected to the OpenEPC through S1 AP interface. Thus, the remote pico base station will not have physical connection to the OpenEPC though it would be the part of the network. The wireless backhaul link and also the VPN connection will act as the backbone connection for the remote base station to be connected to the OpenEPC. The Pico base station operating on 2.6 GHz which is connected to the CPE device via a switch port and separate PC is connected to the switch port only to configure the CPE device. On the other side the TDD base station operating on 3.5 GHz frequency has the wireless backhaul link connection to the CPE device. Thus, the Pico base station will be able to establish S1 connectivity to the OpenEPC through the wireless backhaul link and through the VPN server. The OpenEPC is already implemented in the VMWare workstation where five virtual machines are up and running which are client_UE, EnodeB, SPGW, MME and EPC_Enabler. The VPN server would be installed in the sixth virtual machine which would be integrated with the other virtual machines. Certain topology would be deployed to accomplish the connection between the VPN server and the other virtual machines.

4.1. Customer Premises Equipment (CPE)

The CPE device we are using here is Gemtek model namely WLTFQM-136ACN, LTE B42_43 Cat 6 Single Mode CPE device. The device is supported to LTE band 42 (3,400-3600 MHz). It has IPv4, IPv6 features; It can be operated in NAT mode, Bridge mode, Tunnel mode (GRE, L2TP/L2TP with IPSec, PPTP) or Router mode. It has the feature of DHCP server for IP allocation for the users up to maximum 253 clients supportable.

SCTP wireless protocol is needed for S1 connectivity without tunneling. The CPE device is not supported to SCTP wireless protocols and do not support most of the common tunneling protocols except L2TP and PPTP tunneling protocols. L2TP protocol do not provide any security encryption itself but it can be used with IPSec. Whereas PPTP tunneling protocols have some limitations for having low security standards and lower performance on unstable connections.

The IP Security tunneling is needed to be created starting from the CPE device and ends to the VPN Server. Thus, in such scenario the CPE device will be operated in tunnel mode with L2TP tunneling protocol and on the other end the VPN Server will be configured for L2TP tunneling with IPSec protocol.

The VPN server tunneling protocol need to be matched with the tunneling protocol which we have defined in the CPE device configuration. We found a software VPN named SoftetherVPN which has configurability to use L2TP tunneling protocol along with IPSec.

4.2. SoftEther VPN Server

SoftEther VPN is a powerful VPN software developed by University of Tsukuba, Japan. The VPN software is open source[9] and easy to establish remote access and site to site VPN. It uses the AES 256-bit and RSA 4096- with encryption to provide sufficient security features such as logging and firewall inner VPN tunnel. The VPN Server runs on Windows, Linus, FreeBSD, Solaris and Mac OS X. It is supported to both IPv4/IPv6 dual stack and configurable to all settings on GUI. It has 1Gbps high speed throughput performance with low memory and CPU usage. It is supported to SSL-VPN (HTTPS) and six major VPN protocols (OpenVPN, IPSec, L2TP, MS-SSTP, L2TPv3 and EtherIP) are all supported to VPN tunneling. Android mobiles, iPhones, iPad, Android Tabs, Mac OSX and windows mobiles are compatible for OpenVPN protocols (IP over TCP/UDP) and L2TP/IPSec VPN protocol (L2TP over IPSec) whereas Windows Vista, Windows 7 and Windows 8 are compatible to MS-SSTP VPN protocol (PPP over HTTPS), SoftEther VPN protocol (Ethernet over HTTPS) and L2TP/IPSec VPN protocol (L2TP over IPSec). On the other hand, Linux operating system is compatible for OpenVPN protocols, SoftEther VPN protocols and L2TP/IPSec protocols. Cisco routers or other vendor's L2TPv3 or EtherIP compatible routers can also connect to the L2TP/IPSec SoftEther VPN server. The hardware requirements to run the L2TP SoftEther VPN server is also considerably low. Normally only 2Gbytes of free space in hard drive and very minimal RAM specification is required for the VPN server. On the other hand, for the SoftEther VPN client requirement only 32 Mbytes of RAM is required.

The SoftEther VPN server, namely "vpn5G" is installed in a virtual machine having Linux operating system. The other five virtual machines (Client UE, EnodeB, SPGW, MME and EPC_Enablers) are already installed in the OpenEPC platform created in VMWare vSphere ESXi Server, which is already up and running on arina2. Certain topology will be created to connect the vpn5G to the other connecting virtual machines.

[9] SoftEther VPN Open Source; Available at https://www.softether.org/

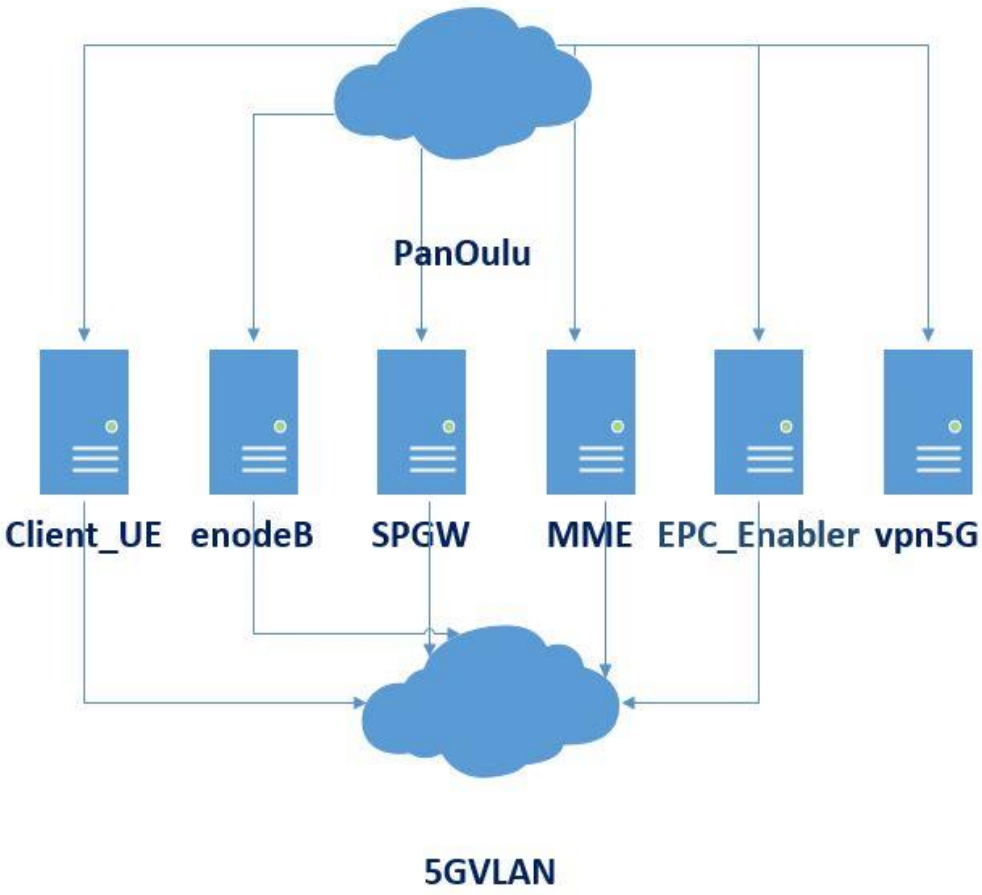

Figure 13. Virtual Machines (VM) in the network

For only security purpose the virtual machine vpn5G only connected to PanOulu and other interfaces which connects to 5GvLAN is disabled, for instance. It is because if all the virtual machines operate in the same network then any configuration change in the vpn5G for any test purpose or for some experiments can in turn change the configuration and topological change, for the other virtual machine may become some technical risks. When the VPN software is fully functional with its configuration, we can easily then connect vpn5G virtual machine to the other virtual machines or even we can clone all the virtual machines including vpn5G and then integrate them together for further measurement and analysis.

To implement such a setup with the vpn5G and the OpenEPC it would be more pragmatic to achieve it step by step implementation where in every step there would be opportunity to deal with the bottlenecks and finding solution and thus continue for further approach to the final implementation. So, we performed the following tests to check and reach the realistic implementation of the final configuration.

4.3. Tunneling between the server and the CPE device

The test is carried out to check if it is possible to create a tunnel between the server and the CPE device. For this we have used two CPE router which are both connected to the TDD base station as given in the figure below:

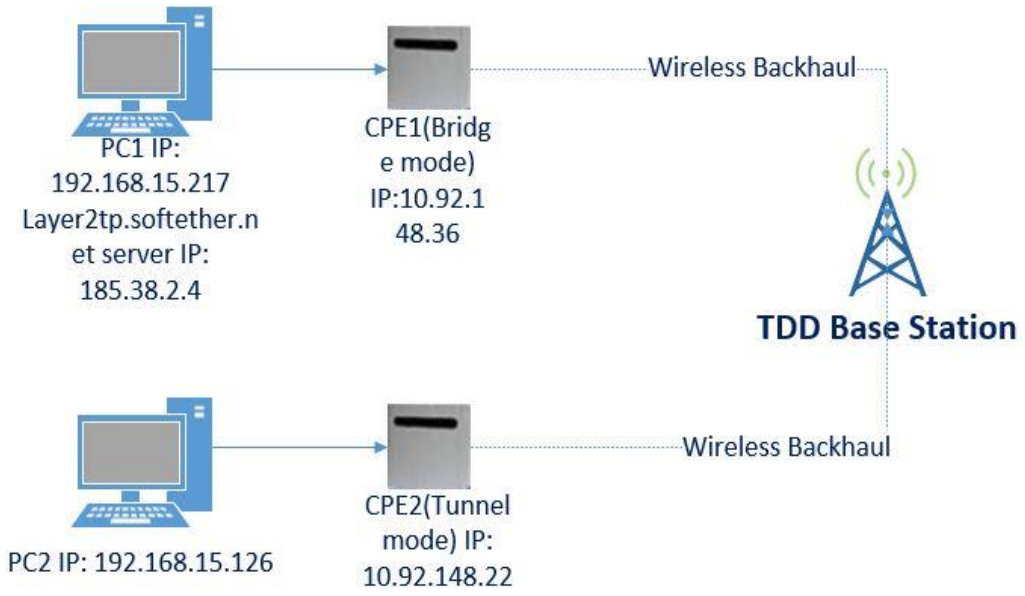

Figure 14. Tunneling between layer2tp.softether server and the CPE2 device

The TDD base station is wirelessly connected with two CPE routers. One CPE is operating on Bridge mode and the other operating on Tunnel mode. The IP address for the bridged CPE1 device is 10.92.148.36 and CPE2 10.92.148.22 which is operating in Tunnel mode. The tunneling has been done between CPE2 and the layer2tp.softether.net server (IP: 185.38.2.4) which is running on a PC, whose IP address is 192.168.15.217. Thus, we are operating CPE1 in bridge mode so that we can ignore CPE1 when we are considering the tunnel between the layer2tp.softether.net server and the CPE2 device. PC2 is connected to the CPE2 device only to configure the CPE device to operate in tunnel mode with other settings and configurations.

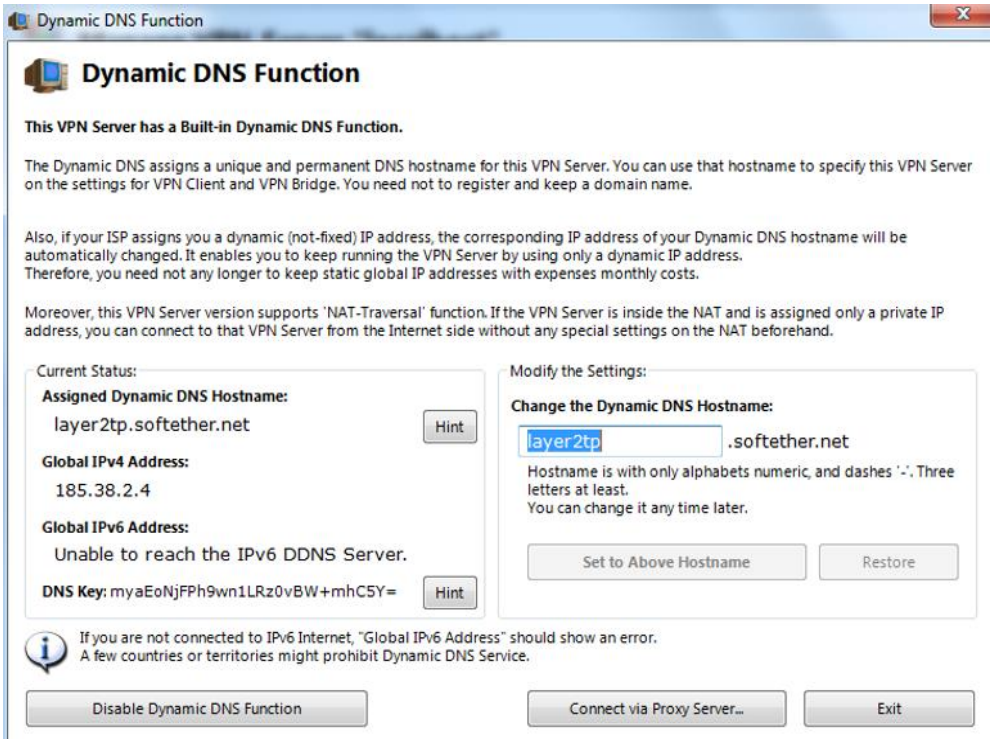

Figure 15.layer2tp.softether.net Server installed in PC1

The VPN Server has been built in dynamic DNS function so that in the software we can customize the name of the VPN server by changing the dynamic DNS hostname. The following figure is the block diagram representation of the figure 1. Here the VPN server IP address is 185.38.2.4 which is installed in a computer whose IP address is 192.168.15.217. The dotted line represents both the CPE devices wirelessly connected to the TDD base station.

4.3.1. layer2tp.softether.net server settings and configuration

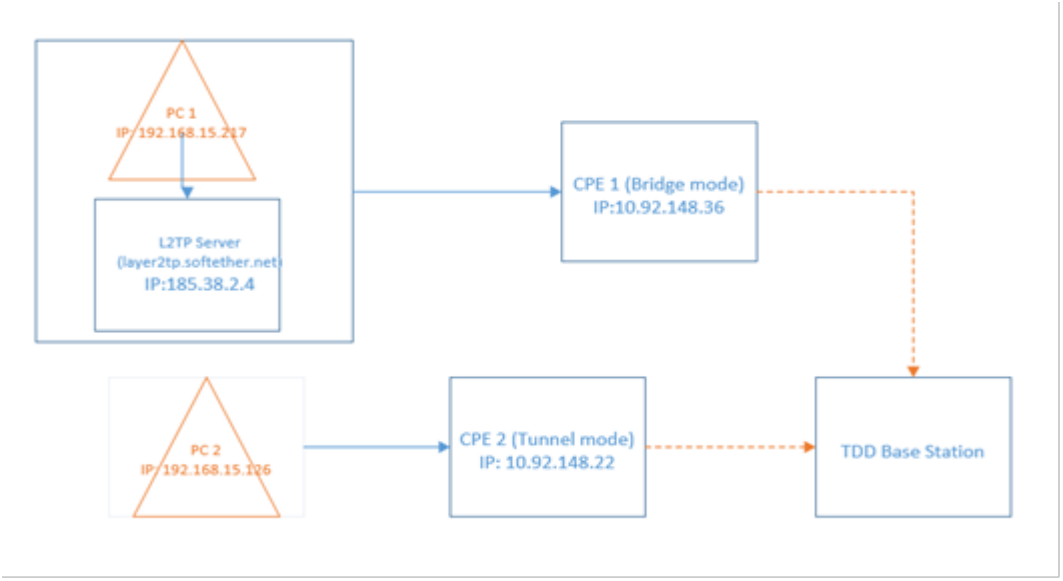

Figure 16. Block diagram representation.

4.3.2. CPE1 and CPE 2 Configuration

The CPE device, a wireless router can be connected to the PC with LAN cable or be wirelessly configured. The configuration setup for the CPE device is simple and user friendly. By providing the IP address (192.168.15.1) and the default user's name and password, it is easy to go to the settings and configure back and forth.

Figure 17. CPE 1 WAN Setup

The CPE1 device name is Generic_91E9E3. The WAN IP address of the device is 10.92.148.3. We have configured CPE1 to operate in bridge mode. The WAN MTU is 1400 and connection mode is here given as DHCP.

Figure 18. CPE 2 WAN Setup

The CPE2 device name is Generic_91EAA3. The WAN IP address is 10.92.148.28. In the WAN settings we have changed the connection mode be DHCP, VPN type as L2TP, supporting NAT have also been configured as CPE2 device to operate in Tunnel mode and the default gateway interface is here given as tunnel. We have given the L2TP server name as the same as has been created in the PC1 which was layer2tp.softether.net. In the L2TP user we have given one of the user's name which we have created in the server in PC1. Here in the settings we provide the IPSec L2TP key as "VPN" which was the same IPSec pre-shared key that we had provided while configuring the VPN server.

We have configured the CPE2 to operate in tunnel mode, acting as a client. So, we can easily check the connectivity by pinging from the client to the layer2tp.softether.net server and vice versa. From the CPE device diagnostic tool interface, we can perform the ping test and trace route.

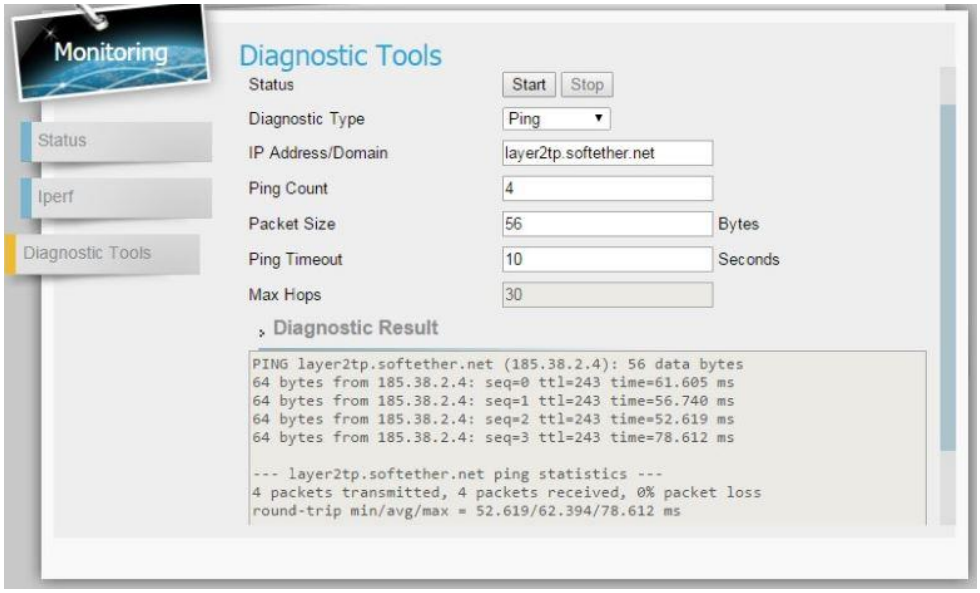

Figure 19. Ping to Server

The above ping test result is found when we ping to the Server from CPE2. Here it is noticeable that the delay is high i.e. 61.605 ms, 56.740 ms and so on.

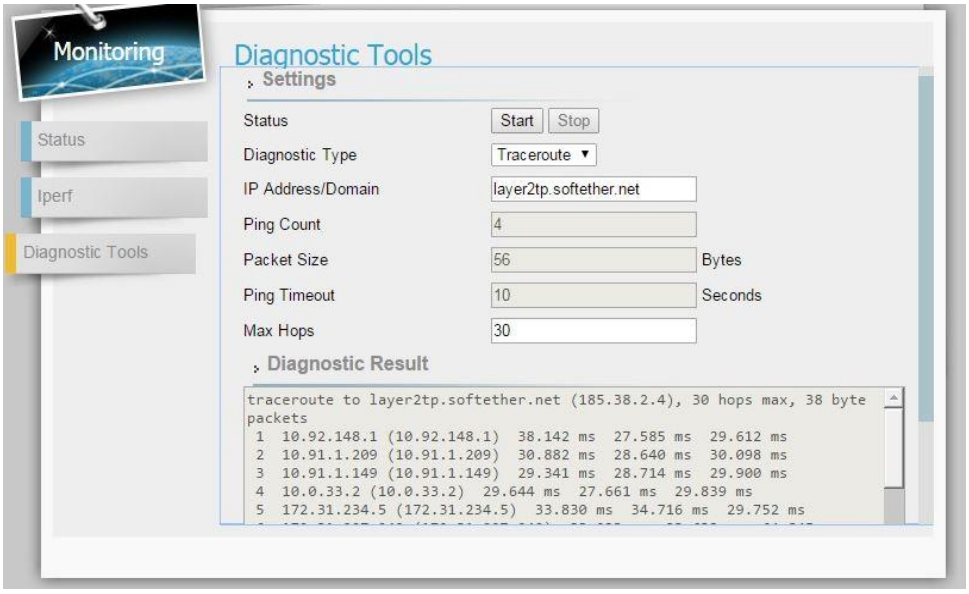

Figure 20. Trace Route to the Server

Thus, the CPE device and the VPN server can be peered to send and receive traffic. From the Client end CPE 2, we can also ping to the Server on its Global IPv4 or Global IPv6 address instead of pinging to the Server name. Similarly, from server end we can ping to the tunnel end i.e. CPE 2. If we need to check end to end connectivity from the Server end to the end device PC2, then we need to install Software VPN client on PC2 so that it can communicate with the server. So, we have established IPSec\L2TP tunneling and passed traffic through the tunnel i.e. from the CPE device to the layer2tp.softetherVPN server in figure 14.

4.4. Tunneling between the PC(client) and the server

Figure 21. Tunneling between client PC and server on the virtual machine

In this arrangement tunneling is configured between the PC(client) and the SoftEther VPN server running it in virtual machine. The SoftEther VPN server is installed in PC running windows server 2012 R2 and the client is configured in a PC which connects with LAN cable. Thus, the server PC and the PC client both are on the 5GVLAN. The server PC IP is 193.166.28.10 and the client PC IP is 193.166.28.56. The server's name is: vpn924935333.softether.net. Global IPv4 address: 185.38.2.3.

We are performing this test to confirm that if we can configure the VPN software in the cloud then we can try to make connectivity from the client.

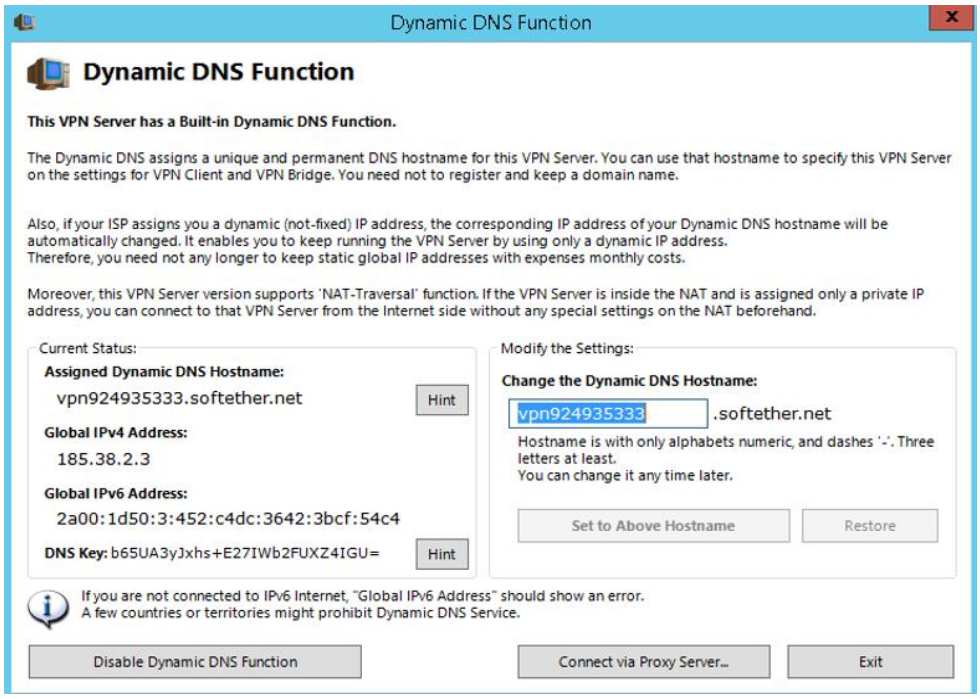

Figure 22. Dynamic DNS Hostname of the Server

This time, we have changed the DNS hostname to vpn924935333.softether.net. The VPN server has both IPv4 and IPv6 addressing which are 185.38.2.3 and 2a00:1d50:3:452:c4dc:3642:3dcf:54c4 respectively.

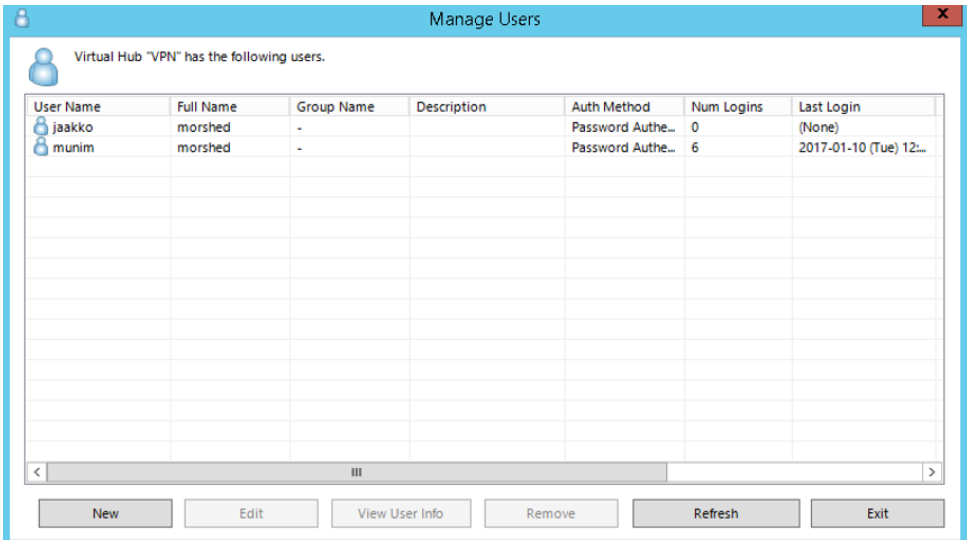

Figure 23. Creating users in the server

We have created users in the server. From the client PC the users can connect to the server. From the client PC, any user between these two users can connect and access the VPN server by providing username and corresponding password.

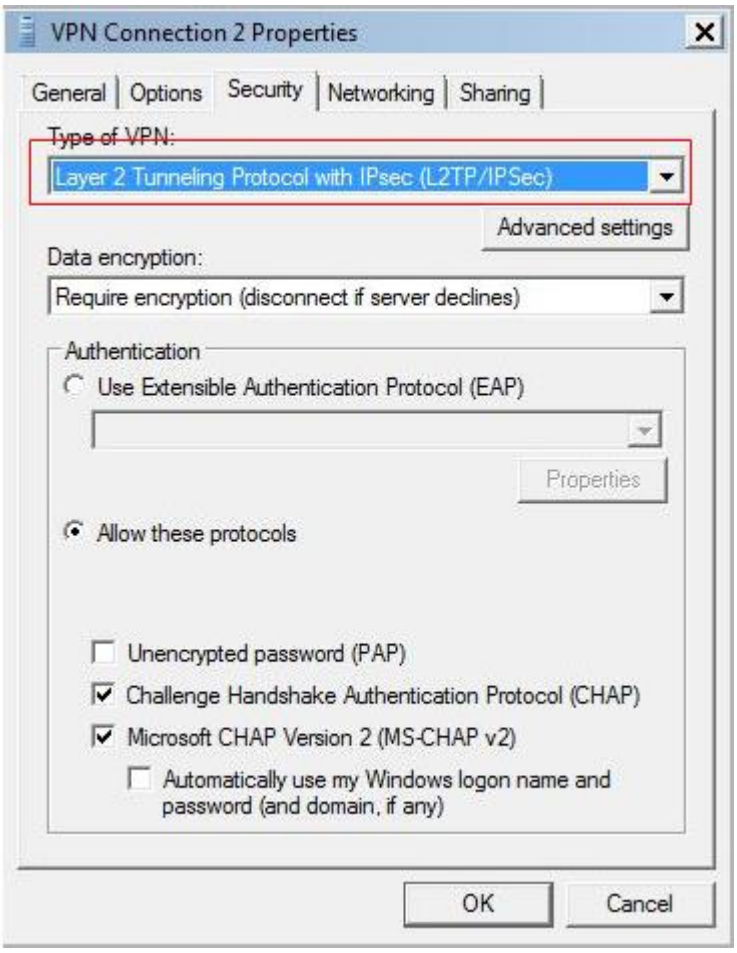

Figure 24. Client tunneling configuration

In the client PC, we have configured the client with layer 2 tunneling protocol with IPSec from the created VPN connection properties. The same secret key need to provide in both VPN server and client to have successful connection between the server and the client.

After successfully connecting the client PC with the server vpn924935333.softether.net, we have performed ping test from the client PC to the vpn924935333.softether.net server.

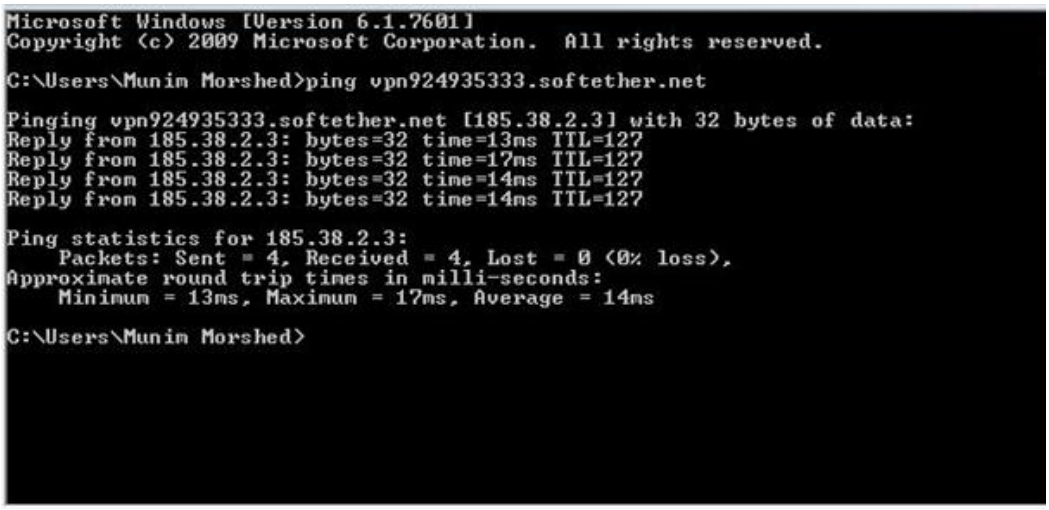

Figure 25. Ping to the server vpn924935333.softether.net

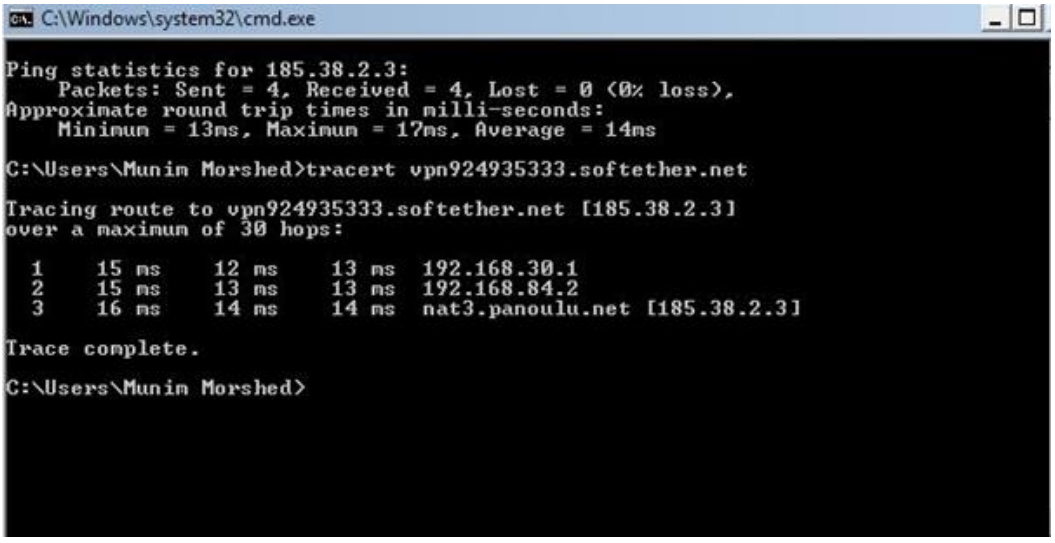

Figure 26. Trace route to the server vpn924935333.softether.net

We have also performed the trace route to the server vpn924935333.softether.net from the client PC.

We can also perform ping or trace route to the global IPv4 address of the server. For example, we can ping to the global address 185.38.2.3, which is the Global IPv4 address of the server.

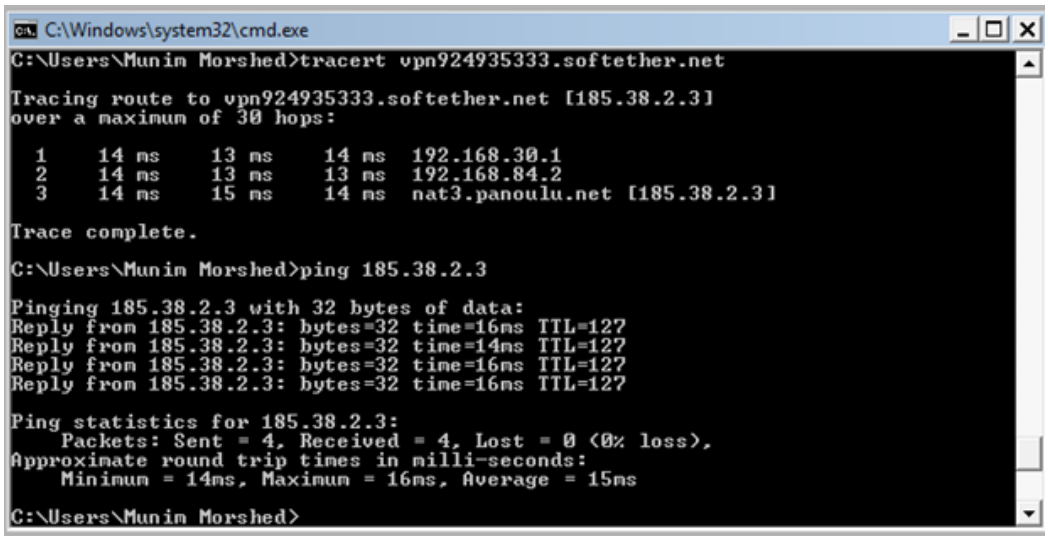

Figure 27. Ping to the server Global IPv4 address

Thus, on a virtual machine running Windows server 2012 R2, we have performed tunneling and data transfer between the Server (vpn924935333.softether.net) and the client PC.

4.5. Tunneling between the CPE device and VPN software

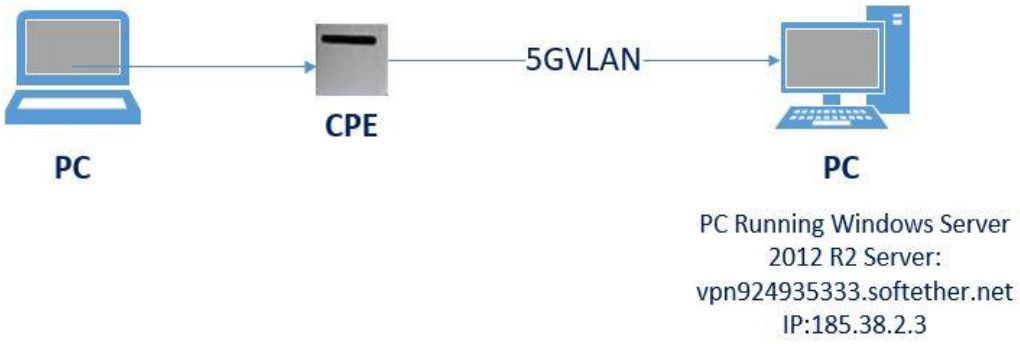

Figure 28. Tunnelling between the CPE device and virtual machine

In this arrangement the tunneling is configured between the CPE device and the VPN software which is running on a virtual machine. Now we can also test the above topology to see if we can connect the CPE device to the SoftEther server which is running on a virtual machine**.** Here this topology CPE device is connected to a PC, which is running windows server 2012 R2. The CPE device and the server PC both are connected on the 5GVLAN. The VPN software vpn924935333.softether.net is installed in the windows server R2 server PC. Another PC is connected to the CPE device only to configure the CPE device.

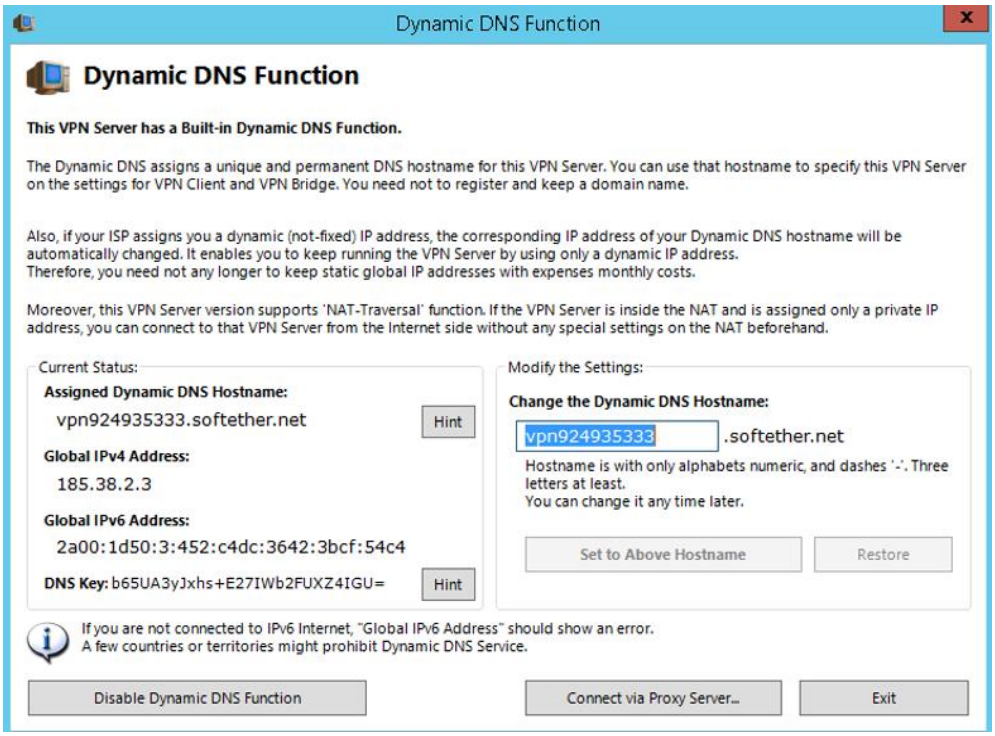

Figure 29. Dynamic hostname of the VPN Server

The VPN server is given the name as vpn924935333.softether.net. The Global IPv4 and Global IPv6 addresses are 185.38.2.3 and 2a00:1d50:3:452:c4dc:3642:3dcf:54c4 respectively.

Figure 30. The WAN Settings of the CPE device

In the CPE device configuration settings, we have changed the WAN settings, LAN settings and other corresponding settings to operate the device in tunnel mode.

As we have configured before we changed the operating mode to tunnel mode, VPN type L2TP, enabling NAT support, default gateway interface as tunnel. We have provided L2TP server name, L2TP user name and password as also IPSec secret key in the WAN settings. Thus, the CPE device will act as a client to the VPN server.

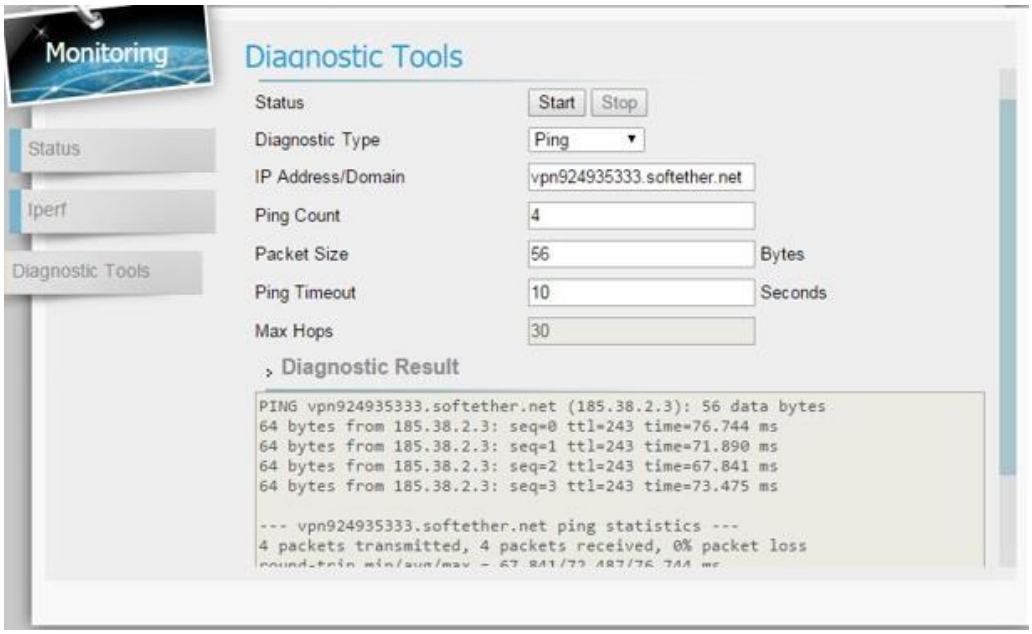

Figure 31. Ping test to the Server vpn924935333.softether.net

From the diagnostic tool interface, we have performed the ping test to the vpn924935333.softether.net server from the client. The packet size is 56 bytes. As we have observed the delay is much higher than usual.

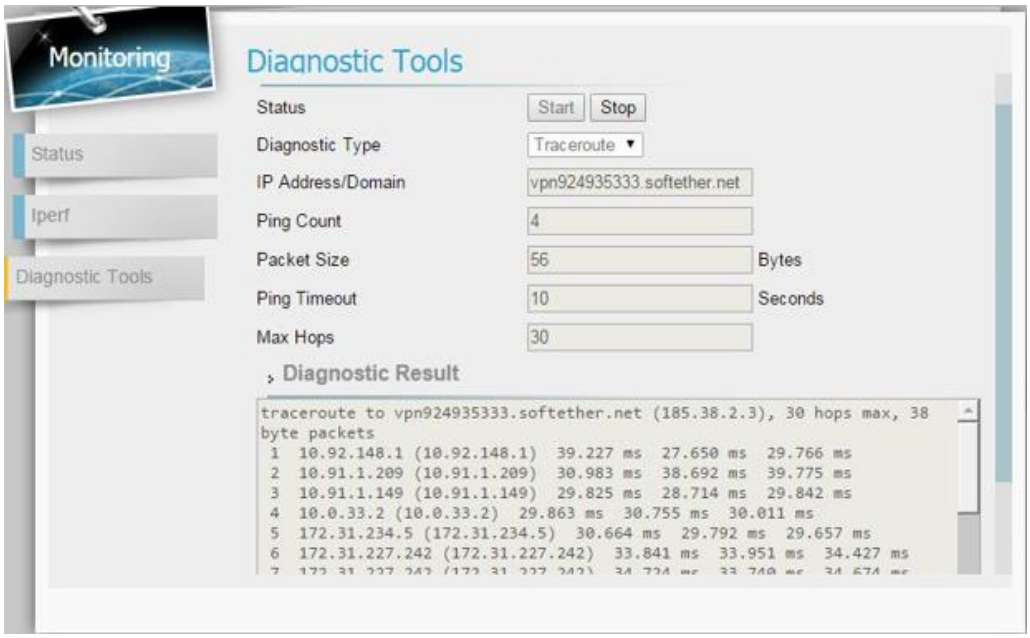

Figure 32. Trace route to the Server vpn924935333.softether.net

To observe the breakdown of delay, we can execute the trace route command from the client to the vpn924935333.softether.net server.

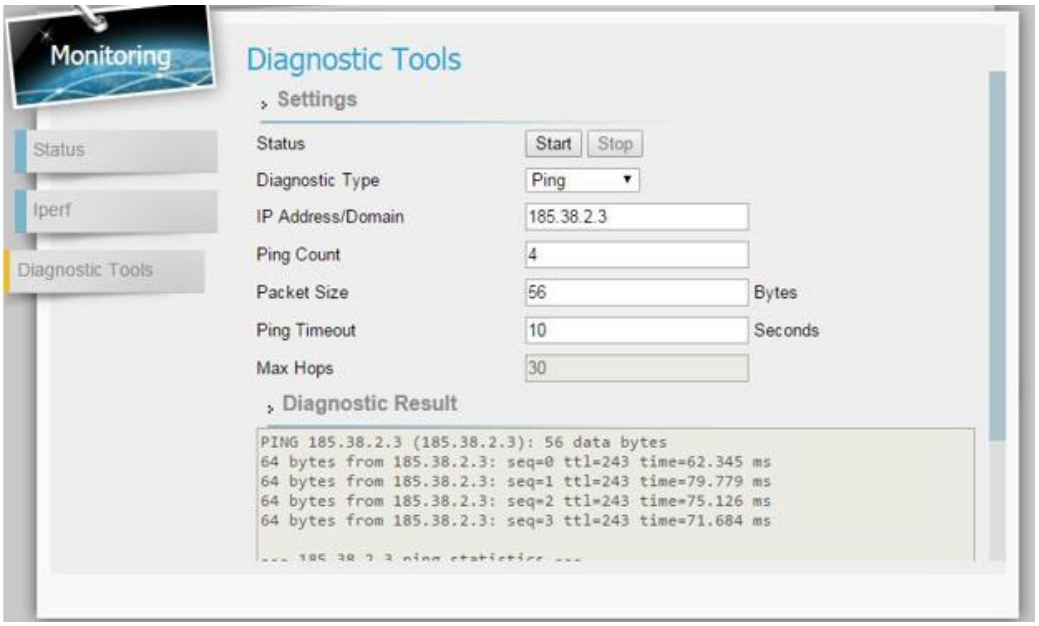

Figure 33. Ping to the server's global address 185.38.2.3 1

We can also directly ping from the client to the destination VPN servers Global IPv4 address which is 185.38.2.3. Pinging to the server's name or pinging to the servers IP address is identical giving the same result. So, After the tunnel establishment we have successfully performed the ping and trace route test from the CPE device to the destination server address.

4.6. Openswan VPN Server

Openswan has been VPN (Virtual Private Network) software for the Linux community since 2005. It has the support for the extensions RFC (Request for comments) and IETF (Internet Engineering Task Force) related to IPSec, including IKEv2, X.509 digital certificates, NAT traversal and many more. Fedora, Red Hat, Debian, Gentoo and many more systems already include the distribution. It is even easy to install by downloading the software directly from the source code. Openswan VPN is much flexible for the users to make changes of the parameters and options to define user's requirements. Openswan supports SSTP, IPSec, L2TP, L2TPv3, PPTP, Split tunneling and SSL/TSL communication protocols[10].

In our system we are using CPE as the client and Openswan as the VPN server. Thus, the client CPE and the VPN server must be compatible with the same protocols. Our CPE device is compatible to L2TP/IPSec and PPTP transport protocols. SoftEther VPN for Linux is also available to be installed. In this thesis we have chosen Openswan in some implementation cases because it is easier to

[10] Welcome to Openswan: Website available at https://www.openswan.org/

implement, finding customer support, fault detection and flexibility in implementation [13].

4.7. Tunneling between PC (client) and L2TP/IPSec Openswan VPN

Figure 34. Tunneling between PC as client and L2TP/IPSec VPN Server

In the server configuration, the tunneling is configured between PC (client) and L2TP/IPSec Openswan VPN running in a virtual machine "vpn5g". The client IP address is defined from 172.16.20.30-172.16.20.100 and the local IP address of the server 172.16.20.1. After tunnel establishment the client PC has the PPP adapter VPN connection with the IP address 172.16.20.30.

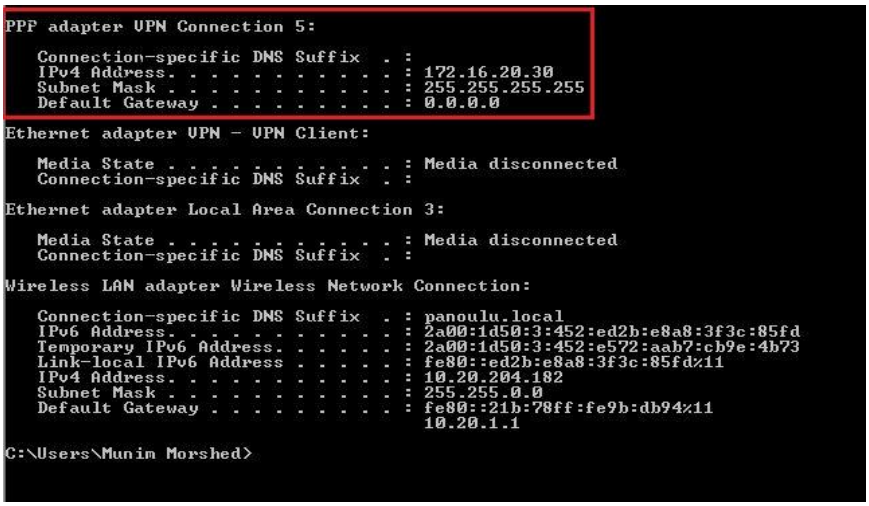

Figure 35. PPP adapter VPN connection between client and server

The VPN connection status can also be viewed from 'VPN connection status' of the client windows PC or it can be viewed from the terminal of Linux terminal of the server. By executing the command '/etc/init.d/ipsec status' from the Linux command line, we can view the number of active established tunnels.

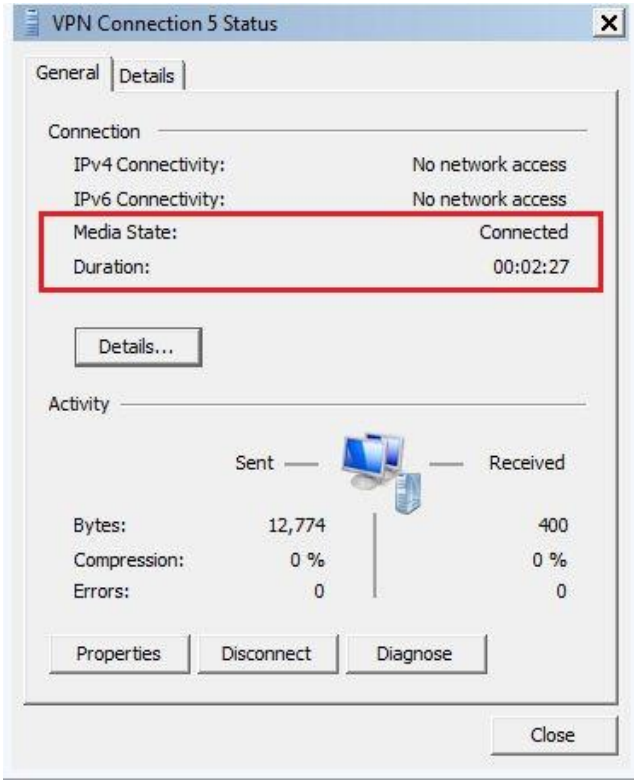

Figure 36. VPN Connection status from Client PC

```
root@epdg:/etc# /etc/init.d/ipsec status
IPsec running  - pluto pid: 16095
pluto pid 16095
1 tunnels up
some eroutes exist
root@epdg:/etc#
```

Figure 37. IPSec Status of the L2TP/IPSec server

Clients can simultaneously connect to the server. For example, an android phone is connected to the server. The client android phone is assigned to an IP address 172.16.20.31 from the server. In the server side the defined IP range for the clients is 172.16.20.30-172.16.20.100.

```
ppp0      Link encap:Point-to-Point Protocol
          inet addr:172.16.20.1  P-t-P:172.16.20.30  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1400  Metric:1
          RX packets:248 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:16912 (16.9 KB)  TX bytes:152 (152.0 B)

ppp1      Link encap:Point-to-Point Protocol
          inet addr:172.16.20.1  P-t-P:172.16.20.31  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1400  Metric:1
          RX packets:14 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:629 (629.0 B)  TX bytes:96 (96.0 B)
```

Figure 38. PP0 and PP1 VPN connection between client and server

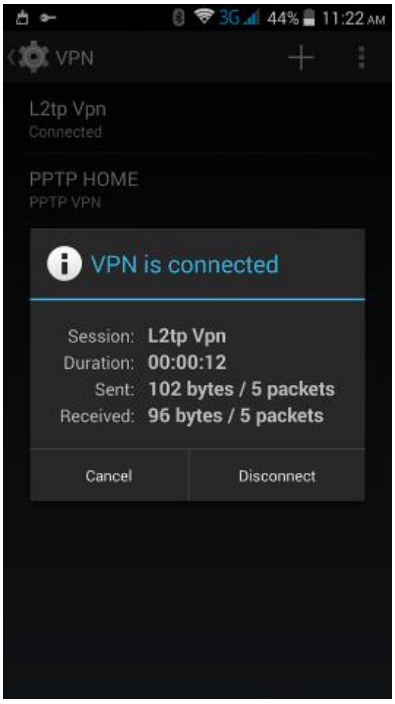

Figure 39. L2TP/IPSec VPN connection between Android client and server

To verify multiple tunnel establishments, we have used an Android phone and one PC as the clients to connect to the server. Figures 38 and 39 shows two point to point connections PPP0 and PPP1 and successful establishment of two tunnels. For every successful IPSec Security tunnel establishment, IPSec SA (Security Association) needs to be established in transport mode. In Linux command line by executing 'tail –f /var/log/auth.log/auth', we can inspect the steps of a successful IPSec SA establishment.

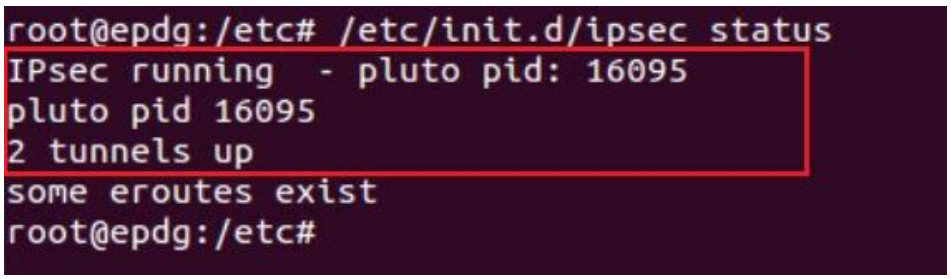

Figure 40. IPSec Status of the L2TP/IPSec server

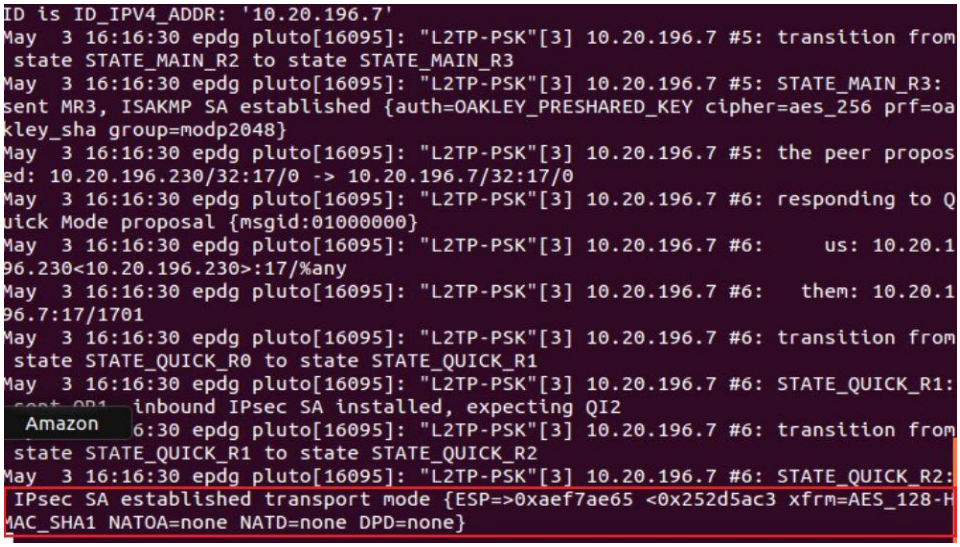

Figure 41. IPSec SA established

Figure 41 shows the IPSec SA establishment in tunnel mode in the log file. After successful tunnel establishment “IPSec SA established transport mode” indicates the final confirmation.

4.8. Tunnelling between PC (client) and PPTP Openswan VPN

Figure 42. Tunnelling between the PC (Client) and PPTP VPN server

In this arrangement the tunneling is configured between PC (client) and PPTP Openswan VPN running in virtual machine vpn5g. In the PPTP VPN server configuration, client IP address is defined from IP 192.168.1.1-192.168.1.100 and the local IP address is defined as 192.168.0.1. Figure 43 and 44 show the successful point to point VPN connection status from the client PC and the server respectively.

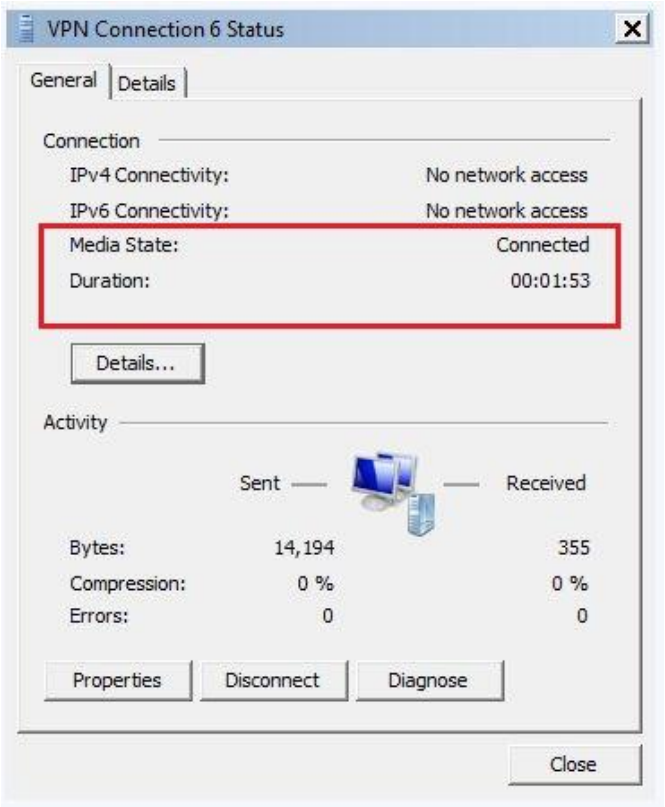

Figure 43. VPN Connection status from Client PC

```
ppp0      Link encap:Point-to-Point Protocol
          inet addr:192.168.0.1  P-t-P:192.168.1.1  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1396  Metric:1
          RX packets:510 errors:0 dropped:0 overruns:0 frame:0
          TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:35979 (35.9 KB)  TX bytes:120 (120.0 B)
```

Figure 44. Point to point VPN connection between client and server

The server successfully is point to point connected to the client 192.168.1.1. From the Linux command line, by executing the command '/etc/init.d/pptpd status' we can check the status of the PPTP server. Figure 45 shows that the PPTP server is up and running.

```
root@epdg:/etc# /etc/init.d/pptpd status
* pptpd is running
root@epdg:/etc#
```

Figure 45. PPTP Server status

4.9. Tunnelling between CPE (client) and L2TP/IPSec Openswan VPN

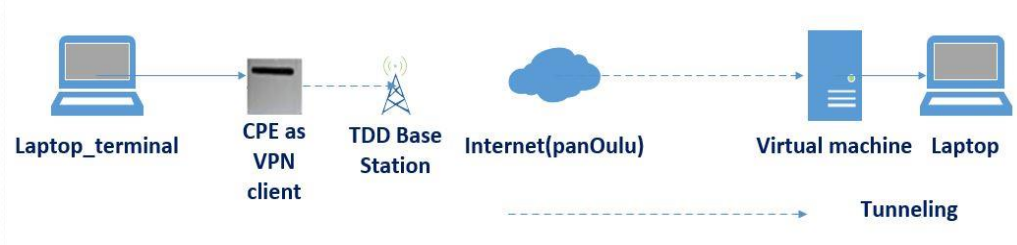

Figure 46. Tunnelling between the CPE (client) and L2TP/IPSec VPN

In the above arrangement, the L2TP/IPSec VPN tunnelling is configured between the CPE and the Openswan VPN running in virtual machine “vpn5g”. The configuration for the CPE device to work as a client, we have changed the WAN setting as following:

Figure 47. CPE device WAN settings for L2TP VPN tunnel mode

The VPN IP address is 10.20.196.230. In the server settings we have created a user “student” with password. In the Client CPE configuration, we have also given

the username and corresponding password accordingly. The CPE device has a built in Host name for the device i.e. “Generic_3B7DD”.

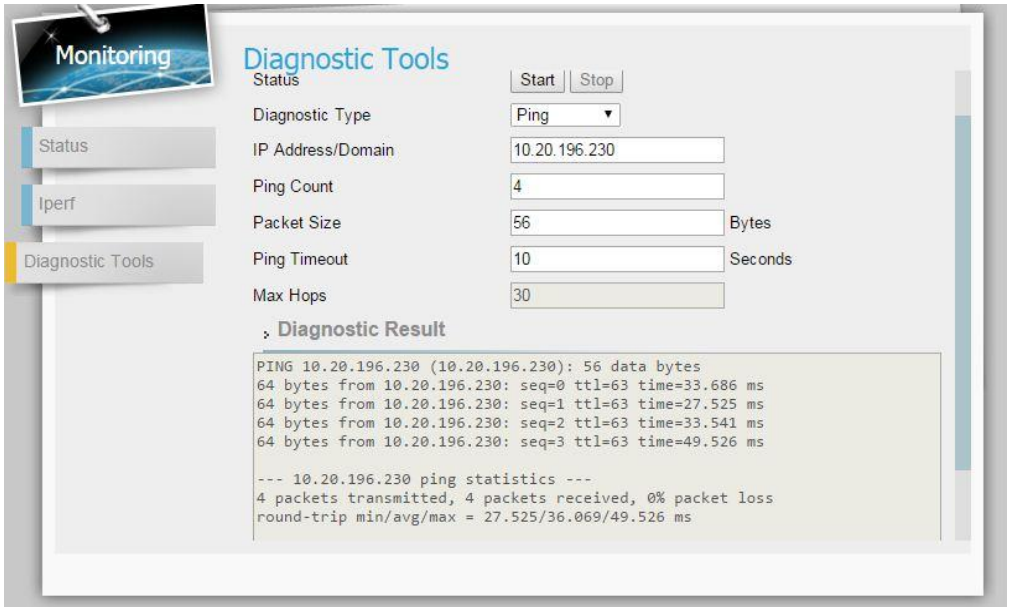

Figure 48. Ping test from CPE (client) device

We have successfully pinged to the server from the CPE device. We have used the diagnostic tools of the CPE device to execute the ping command. It is observed that the delay is significantly high.

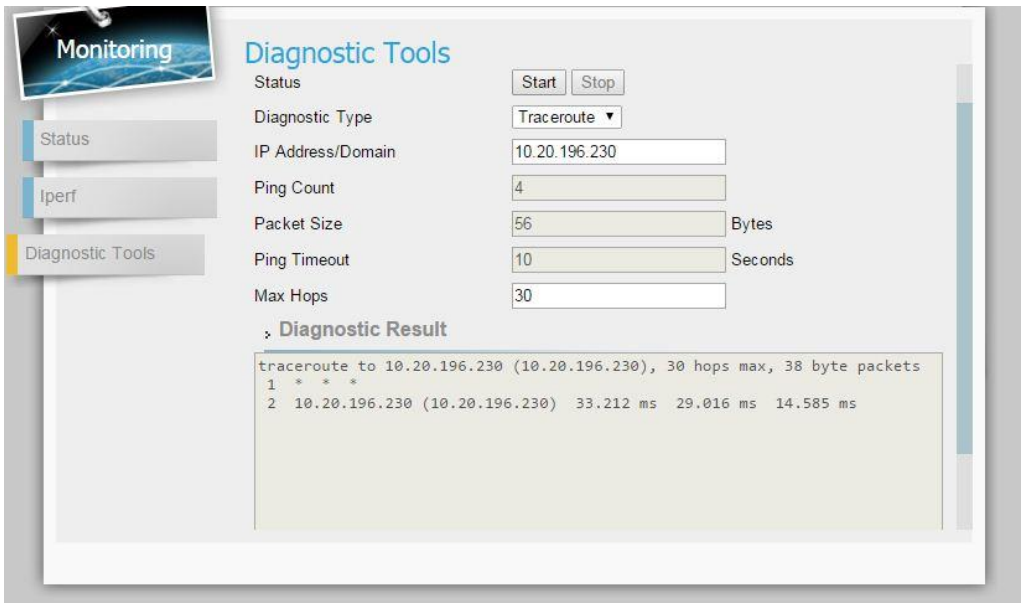

Figure 49. Traceroute to the server from CPE device

We have also performed traceroute to the server from the client CPE device. It is also observed that the delay is also significantly high. We can check for the system log by executing the command ‘tail/ -f /var/log/syslog’ in the command line.

```
root@epdg:~# tail -f /var/log/syslog
May  3 14:28:14 epdg dhclient: bound to 10.20.196.230 -- renewal in 786 seconds.
May  3 14:28:14 epdg NetworkManager[1208]: <info> (internet): DHCPv4 state change
d renew -> renew
May  3 14:28:14 epdg NetworkManager[1208]: <info>   address 10.20.196.230
May  3 14:28:14 epdg NetworkManager[1208]: <info>   prefix 16 (255.255.0.0)
May  3 14:28:14 epdg NetworkManager[1208]: <info>   gateway 10.20.1.1
May  3 14:28:14 epdg NetworkManager[1208]: <info>   nameserver '10.20.110.4'
May  3 14:28:14 epdg NetworkManager[1208]: <info>   nameserver '10.20.110.5'
May  3 14:28:14 epdg NetworkManager[1208]: <info>   domain name 'panoulu.local'
May  3 14:28:14 epdg dbus[460]: [system] Activating service name='org.freedesktop
.nm_dispatcher' (using servicehelper)
May  3 14:28:14 epdg dbus[460]: [system] Successfully activated service 'org.free
desktop.nm_dispatcher'
```

Figure 50. System log information

The ping command from the CPE device can also be traced from Wireshark packet analyser tool. The IP address for the CPE device and the L2TP/IPSec server is 10.20.208.163 and 10.20.196.230 respectively. By filtering ICMP files we can trace the source and destination of the traced file.

21879	193.64904906	10.20.208.163	10.20.196.230	ICMP	100 Echo (pi
21880	193.64910006	10.20.196.230	10.20.208.163	ICMP	100 Echo (pi
21895	193.77899506	172.16.20.31	4.2.2.1	DNS	123 Standard
21898	193.78396506	172.16.20.31	8.8.4.4	DNS	123 Standard
21923	193.92395106	172.16.20.31	4.2.2.1	DNS	114 Standard
21987	194.66917706	10.20.208.163	10.20.196.230	ICMP	100 Echo (pi
21988	194.66920206	10.20.196.230	10.20.208.163	ICMP	100 Echo (pi
22094	195.67910306	10.20.208.163	10.20.196.230	ICMP	100 Echo (pi
22095	195.67913706	10.20.196.230	10.20.208.163	ICMP	100 Echo (pi
22179	196.67908306	10.20.208.163	10.20.196.230	ICMP	100 Echo (pi
22180	196.67912506	10.20.196.230	10.20.208.163	ICMP	100 Echo (pi
22300	197.78900106	172.16.20.31	4.2.2.1	DNS	123 Standard
22303	197.80943306	172.16.20.31	8.8.4.4	DNS	123 Standard
22368	198.79906706	172.16.20.31	4.2.2.1	DNS	123 Standard
22371	198.81911606	172.16.20.31	8.8.4.4	DNS	123 Standard
22385	198.92904206	172.16.20.31	8.8.4.4	DNS	114 Standard
22458	199.79909006	172.16.20.31	4.2.2.1	DNS	123 Standard
22463	199.81893906	172.16.20.31	8.8.4.4	DNS	123 Standard
22671	201.80902306	172.16.20.31	4.2.2.1	DNS	123 Standard
22674	201.81903406	172.16.20.31	8.8.4.4	DNS	123 Standard

Figure 51. Wireshark trace for the ping request

4.10. Tunnelling between CPE (client) and PPTP Openswan VPN

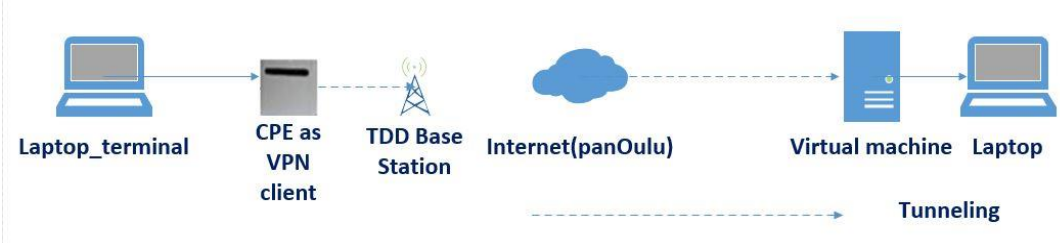

Figure 52. Tunnelling between the CPE (client) and PPTP VPN server

In this configuration the PPTP (point to point tunnelling protocol) is used in the VPN server, while the CPE working as a client. In the client CPE configuration, we have changed the PPTP username and password accordingly. Similarly, like L2TP server, we have changed settings for PPTP VPN type. The PPTP Server IP address is 10.20.196.230. The PPTP client CPE WAN configuration is as follows:

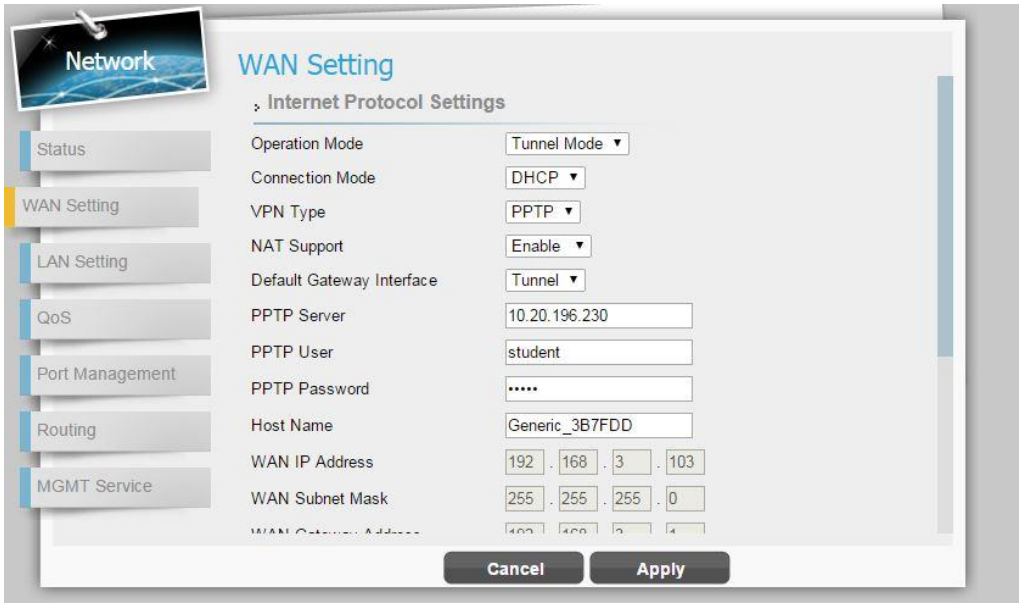

Figure 53. CPE device WAN settings for PPTP VPN tunnel mode

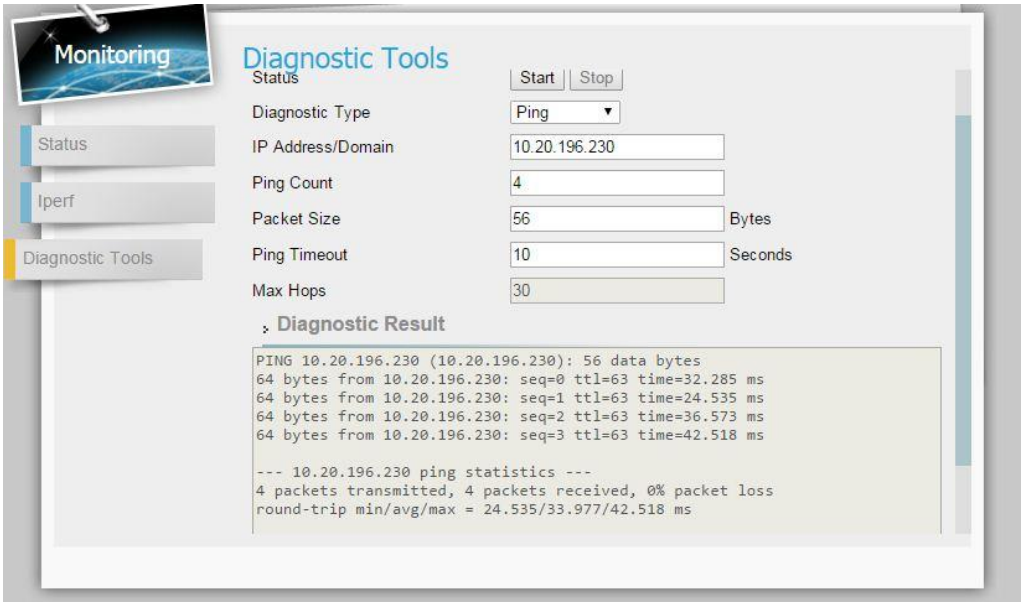

Figure 54. Ping test from CPE (client) device

We have executed ping command from the client CPE to the PPTP VPN server. It is also observed that the delay is significantly high.

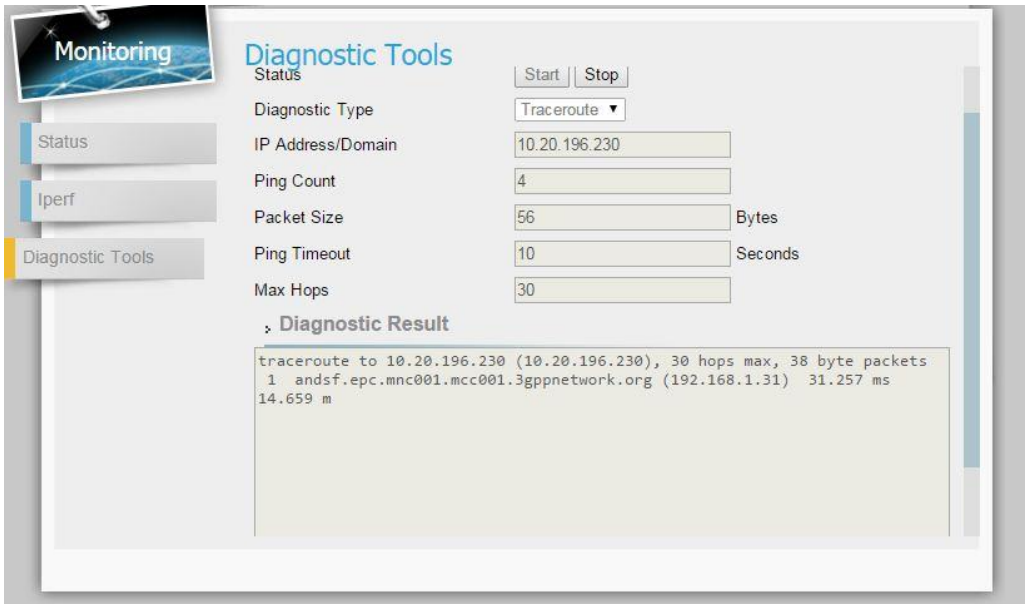

Figure 55. Traceroute to the server from CPE device

We have also executed the traceroute command from the client CPE to the PPTP server IP address. It is also observed that the delay is significantly high.

4.11. StrongSwan VPN Server

StrongSwan VPN is the IPSec implementation which runs on Linux 2.6, 3.x and 4.x kernels, Android, FreeBSD, OS X, iOS and windows. It's the open source IPSec based VPN solution which can be implemented for both IKEv1 and IKEv2 key exchange protocols. StrongSwan is supported to IPv6 IPSec tunnel, dynamic IP address allocation and interface update, NAT- Traversal via UDP encapsulation and port floating, dead peer detection etc. The virtual IP address pool is managed by IKE daemon/SQL database. It has secure IKEv2 EAP user authentication, supported to EAP-RADIUS plugin and IKEv2 multiple authentication exchange. The authentication procedure is based on X.509 certificates or pre-shared keys[11]

LibreSwan is a fork of OpenSwan. FreeSwan project is the mother/origin of the both strongSwan and LibreSwan. OpenSwan or LibreSwan is much similar or close to the origin where as StrongSwan is a complete reimplementation in some criteria.

[11] StrongSwan VPN server configuration details available at https://www.strongswan.org/

4.12. Tunnelling between CPE(Client) and L2TP VPN Server

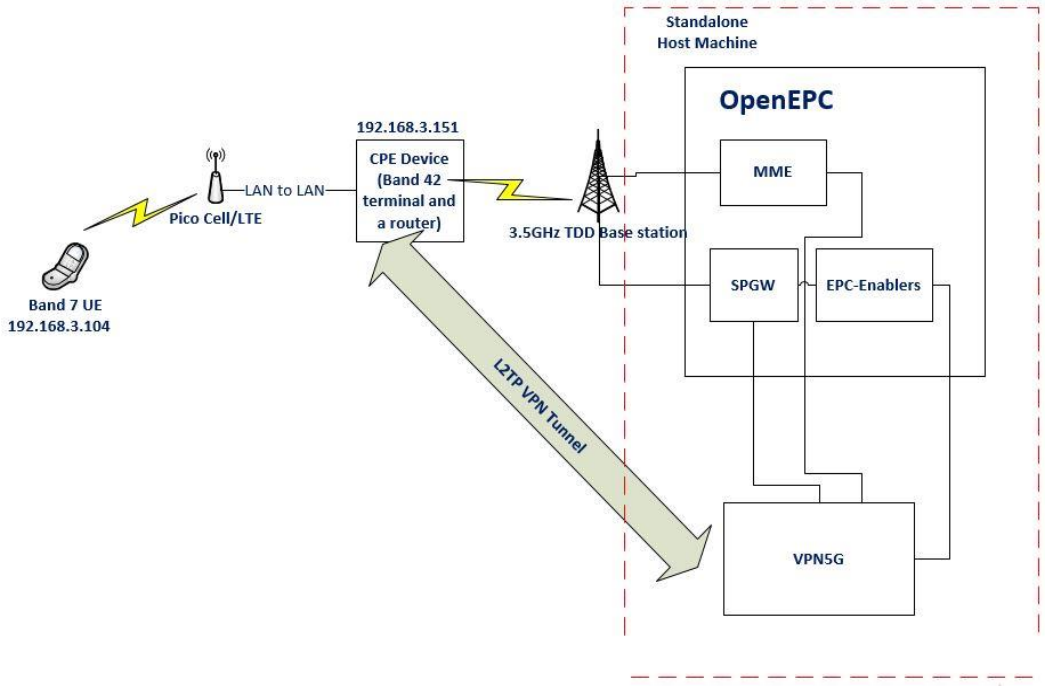

Figure 56. Stand-alone architecture of the setup

The standalone setup implementation of the network is shown in the above figure 56. In the setup CPE device is configured to operate in tunnel mode. OpenEPC SIM card is used in both CPE device and UE. The 'vpn5g' virtual machine along with the other virtual machines are running on top of hypervisor which is the Vmware workstation installed on the host computer. StrongSwan L2TP IPSec VPN is installed in the VM 'vpn5g'. The WAN settings of the CPE is shown in the following figure 57. The IPSec secret and other operational mode settings are provided to the CPE device settings. We have given a static IP to the CPE and Pico LAN interfaces initially which are 192.168.200.100 and 192.168.200.150 respectively to bring the devices in the same network. In the Pico base station settings, we define CPE device as the gateway for the Pico to enable successful traffic from the Pico to OpenEPC. The Pico base station is operating on 2.6 GHz and the TDD base station on 3.5 GHz frequency. In the VPN server configuration, we have configured the local IP address for the server as 192.168.4.15 and the client range is defined as 192.168.4.16-17.

In the host computer we have used two ethernet ports. One port is bridged to the TDD base station and the other is bridged with the 5GvLAN. The host IP interface of 5GvLAN is 193.166.28.150. The host computer is connected to the internet via 5GvLAN. All the OpenEPC virtual machines/nodes i.e. EPC-enablers, MME, SPGW etc along with 'vpn5g' are running on the Vmware workstation which is running on the host computer. The VPN VM has interface with an IP 193.166.28.151 on the 5GvLAN IP pool. This IP 193.166.28.151 is the external IP address for the VPN machine.

The CPE, working as a client having the LAN IP 192.168.200.100 is paired with the VPN server whose external IP address is 193.166.28.151. The CPE WAN settings are given in the following figure 57.

Figure 57. WAN settings of the CPE device

The CPE and the VPN server are given the same user name and password for authentication. The layer 2 tunneling protocol request reaches the external IP of the VPN server. All necessary settings along with Unix authentication is required in the VPN server. In the VPN server settings, we have defined the client IP range as 192.168.4.16-17 and the local IP address of the VPN server as 192.168.4.15. In the VPN server, the DNS IP 4.2.2.1, we have used chap authentication in both client and server machines. The details of the configurations are included in the appendix section.

Figure 58. PPP0 established in L2TP tunnel mode

The CPE has successfully been attached to the core network. From the CPE diagnostic tool, we can ping successfully to the MME IP 192.168.4.80 and to the google DNS 8.8.8.8. To accomplish this connectivity, we have imposed a SNAT rule on the iptables of the VPN machine. The WAN IP of the VPN machine is 192.168.3.151

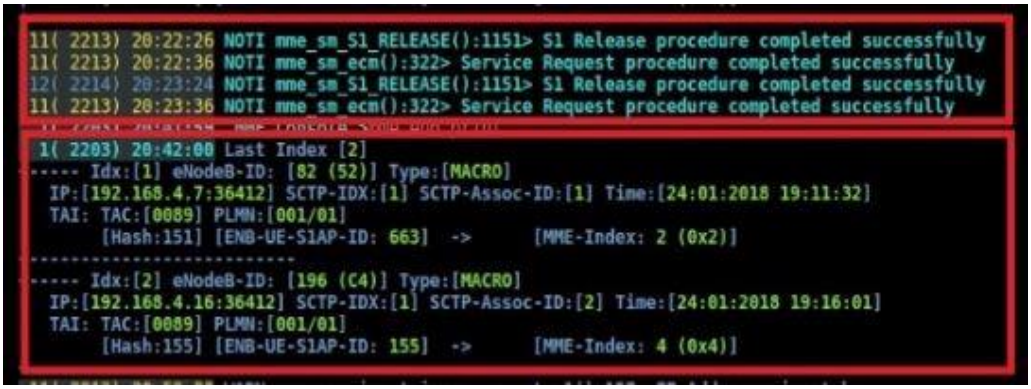

Figure 59. Base station attachment procedure

The pico base station, having the gateway IP 192.168.200.100 (CPE), is successfully attached to the network. The attachment procedure messages are shown in the figure 59. From the command line “mme.enb.print” we can view the details of both the base stations. The TDD base station IP address is 192.168.4.7 and the pico baco base station IP address is 192.168.4.16.

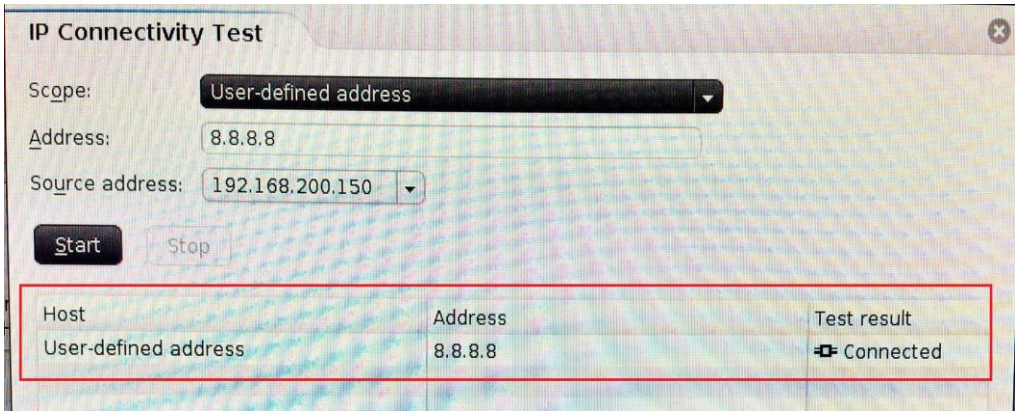

Figure 60. Connectivity test from pico base station to 8.8.8.8

From the pico base station terminal we can successfully check the connectivity to MME (192.168.4.80) and to the google dns 8.8.8.8 and 4.2.2.1 for internet connectivity.

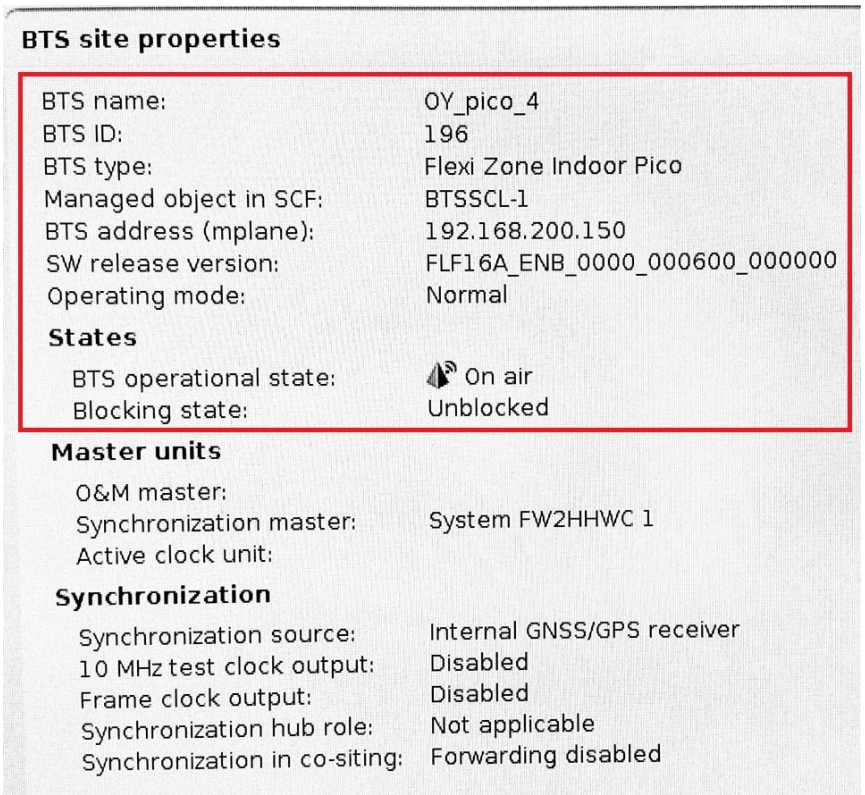

Figure 61. Pico base-station on Air

To make the pico base station reachable, it was required to give “IP forwarding” commands in the VPN machine. The local IP address of the pico base station is 192.168.200.150. In the pico base station configuration, we have given the gateway is the CPE device i.e 192.168.200.100 rather than MME IP address for other usual cases.

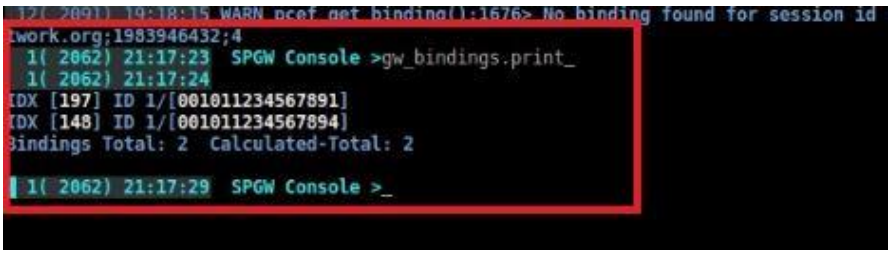

Figure 62. SPGW binding information

The pico base station attached with the network. From the Nokia BTS site manager tool, we can check the state of BTS, available alarm, local IP and other network configurations. The binding messages are generated and can be viewed for detailed information. The base station needs to be synchronized for successfully going to “On Air”.

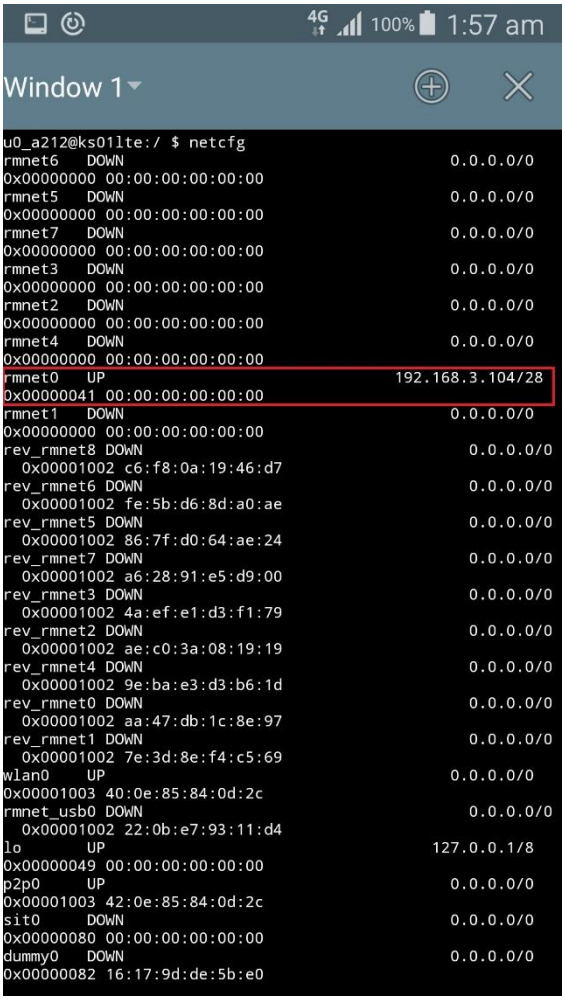

Figure 63. rmnet0 interface up in UE

According to the openEPC configuration settings, UE is expected to have an IP on the nat_c IP pool i.e 192.168.3. x. In figure 63, the UE is successfully assigned to an

IP on the nat_c IP pool. According to LTE assignment, the UE gets the IP 192.168.3.104 on the rmnet0 interface.

5. DISCUSSION

The future of cellular communication is assumed to be the integration of huge number of connecting devices with very high throughput capabilities. Researchers all around the world are working to meet the requirements. All the efforts, extensive measurements and research initiatives are imposed in the research and development of different industries, institutes and research body of academia are working side by side with a common aim to materialize the speculations to bring them into reality. In general, 5G technology must be more efficient, stable, more scalable and compatible platform to integrate all previous technologies. 5G technology is expected to have huge devices, IOT and sensors connected in the network. In most of the cases, the connections are by wireless which is expected in future cellular technology.

As IPv4 addresses are being exhausted researchers implement IPv6 on carrier network. IETF has proposed several technologies for smooth transition from IPv4 to IPv6. These technologies can be divided in three types: Dual stack, tunnelling and translation. In dual stack technology IPv4 and IPv6 protocol stacks are implemented in both hosts and routers so that host can use either one of IPv4 or IPv6 addresses to access the outside networks. In tunnelling technology, the most important part lies on the encapsulation and decapsulation of IPv4 and IPv6 packets according to network types. The solution can be discussed on two different scenarios. In first scenario, it is possible to implement L2TP 6in4 gateway mode or L2TP 6in4 host mode. In L2TP 6in4 gateway mode, host in an IPv4 LAN need to access IPv6 network through a gateway where as in L2TP 6in4 host mode a host in IPv4 network wants to access IPv6 network directly. In this 6in4 host mode the host acts as LAC. In second scenario we can implement L2TP 4in6 gateway mode or L2TP 4in6 host mode. In L2TP 4in6 gateway mode hosts in an IPv6 internet can access IPv4 network through LAC acted as gateway whereas in L2TP 4in6 host mode a host in IPv6 only network wants to access the IPv4 only network directly. Here the host of the L2TP 4in6 host mode acts as a LAC [14].

L2TP protocol is used on authentication security of WAN. Authentication and reliability of transmission can be improved by using a double-sided authentication and similarity mppe encryption. WLAN security authentication has WEP, WPA and 802.1x pattern.

WEP pattern has few security vulnerabilities WPA pattern an authentication mode of 802.11 standards has two ways: i. pre-shared key pattern and ii. Authentication key publication manner IEEE 802.1x. WPA manner strengthened the algorithm of encryption key, adding features to prevent data from tempering and certification function. WPA pattern has drawbacks such as lack of authentication from tempering on the half way, not resisting the generalized packets and complicate configuration. 802.1x authentication relies on Extensible Authentication protocol EAP. The authentication mechanism requires a dedicated RADIUS server, relatively high installation and maintain cost. It has drawbacks such as lack of integrity protection and data encryption while transmission, possibility of man in the middle attack, session hijacking and denial of service attack. Double-sided L2TP protocol authentication method has four times handshake mechanism MS-CHAPv2 is

considered as most appropriate for authentication of link layer tunnel technology. The encryption method achieved same efficiency of ppp encryption in L2TP. An enhancement in the reliability transmission accomplished since every packet contains the identity information of LAC, LNS and hidden authentication.

L2TP provide error and flow control and the use of ppp level data encryption for the data security and encryption. The data security can be further enhanced by adopting other encryption technologies combined with the L2TP. Examples of such encryption technologies are SSH, SSL, IPSec etc increase the delay no more than 2ms and capable of debugging in Linux systems [15].

As an example, we can consider a scenario where a network consists of main campus and several remote networks. The main campus is connected to the remote networks with VPN technology. IPSec VPN technology considered to be more suitable for remote multi-campus remote interconnection, while L2TP VPN is more suitable for mobile users using remote connection. To achieve more secured and complete remote experience, mobile users can utilize L2TP over IPSec VPN to connect the main campus while having data encryption and enhanced security. To enhance more security firewall can be deployed to the main campus network to achieve safe interconnection between the main campus and the sub campus. IPSec VPN, L2TP over IPSec VPN and firewall technology can be deployed for improved experience for the remote access users to the main campus. Successful implementation of such kind of network can be utilized in colleges, universities and enterprises for secure band complete access experience [16].

For the continuous growing need of the user for more data speed and connectivity without substantial financial investment, NFV and SDN has become the only solution for future cellular communication. NFV based solution comprises of software modules of the core network functions. These software modules are running on separate virtual machines on private cloud or cloud owned by companies for example amazon.com. NFV has virtualized core network components running on virtual machines in the data centres. On the other hand, SDN based solution has the SDN controller where the control function runs as applications. Here the control plane consists of one or more controllers which is considered as the brain of the SDN network [17].

My thesis includes connecting a base station wirelessly to the core (OpenEPC) by using a opensource VPN software. The LTE wireless backhaul is used to achieve the S1 connection to the core network. Normally we cannot make a S1 connection within another S1 connection. The introduction of the VPN solved the issue in a way that the packets come from the Pico encapsulated in the CPE(client) device then using the wireless 3.5 GHz LTE link of TDD base station, finding the gateway of OpenEPC (spgw) coming to the interface of the VPN machine. In the VPN machine the packets get de-encapsulated. The VPN has its connectivity to the other VM's of the OpenEPC. Thus, it appears as a separate base-station traffic coming to the core network. (Figure 56). The procedure makes possible the S1 connection over the air interface.

5GTN is a testbed for all research group to come up with ideas and implement their innovative thinking bringing them into reality. In my thesis work, primarily we have planned to use L2TP and PPTP tunnelling protocol to create the tunnel by which the traffic can pass through. We have tried to reach our goal step by step, solving the bottlenecks in every stage of the implementation. As we have discussed earlier, we ended up deciding that the implementation would be a stand-alone setup (Figure 56).

Stand-alone setup bring ease in creating the network interfaces. The OpenEPC developed by CND (Core Network Dynamics) do not have the repository to create a separate VM with the required interfaces. To overcome the problem, we have manually created a VM “vpn5g” and the required interfaces to connect with the other nodes of OpenEPC.

At the present scope of implementation, the CPE(client) must be in the coverage of the TDD base station. This constraint can be avoided if we configure a global server (http server) and replace the existing local server. Using the global server, we can connect the pico base station from anywhere and within any network ambience. This modification of work would be the future go of the future thesis.

In stand-alone implementation, we have used L2TP tunnelling protocol to establish the tunnel. In extension of this work, we can implement the same setup using PPTP and GRE tunnelling protocol and check for the performance in every case.

In our implementation, we have only used one single pico base station. We can extend this implementation by connecting several base stations (Figure 3) and evaluate the network behaviour, looking for any performance drops, abnormalities as well as evaluating all performance metrics end to end all the way.

6. SUMMARY

For the growing need of coverage and capacity higher frequencies are going to utilized for communication. The cell size would be reduced in size and increased in number. To deploy and achieve such infrastructure with wired backhaul is expensive and tiresome work for some distant remote location. Therefore, the deployment of wireless backhaul can be a good option for network operators to deliver coverage and capacity for high subscriber density areas with reduced cost. Wireless backhaul optimization, performance and scalability will be on the critical path on such cellular system deployment. We have used a software VPN to connect a pico base station using wireless backhaul. The network is using 3.5 GHz wireless link instead of LAN wire for backhaul link between the EnodeB and the core network (OpenEPC). LTE TDD band 42 acting as a wireless backhaul (Link between EnodeB and band 42 CPE router). The VPN tunnel status, UE attachment procedure, CPE router attachment procedure and data packets are traced using Wireshark at different nodes of network. We have tried to implement the whole setup step by step towards the final standalone setup. The standalone setup (Fig.56) having a number of virtual machines with LINUX OS to run the network instances of OpenEPC. The implementation and bottlenecks are documented in this thesis work. Connecting a base station using VPN technology gives the mobility for the BS to provide coverage and capacity to remote distant locations. This plug and play use of network device brings easy and mobile use of base stations. The extension of this work can be done using a global VPN so that the CPE acting as a client can connect to the server from anywhere to establish the tunnel. The pico base station which is connected to the CPE through ethernet port can pass through the tunnel to reach MME for authentication.

7. REFERENCES

[1] Arif, M., 2017. OPENEPC INTEGRATION WITHIN 5GTN AS AN NFV PROOF OF CONCEPT.

[2] Huq, K.M.S. and Rodriguez, J. eds., 2016. *Backhauling/fronthauling for Future Wireless Systems*. John Wiley & Sons.

[3] Zhang, Z., Wang, X., Long, K., Vasilakos, A.V. and Hanzo, L., 2015. Large-scale MIMO-based wireless backhaul in 5G networks. *IEEE Wireless Communications*, *22*(5), pp.58-66.

[4] Lakbabi, A., Orhanou, G. and El Hajji, S., 2012, December. VPN IPSEC & SSL technology Security and management point of view. In *Next Generation Networks and Services (NGNS), 2012* (pp. 202-208). IEEE.

[5] W. Townsley and A. Valencia, "Layer Two Tunneling Protocol L2TP" (RFC 2661), August 1999

[6] Kara, A., Suzuki, T., Takahashi, K. and Yoshikawa, M., 2004, September. A DoS-vulnerability analysis of L2TP-VPN. In *Computer and Information Technology, 2004. CIT'04. The Fourth International Conference on* (pp. 397-402). IEEE.

[7] Hu, M., Zhao, Q., Kuramoto, M., Cho, F. and Zhang, L., 2011, October. Research and implementation of layer two tunneling protocol (L2TP) on carrier network. In *Broadband Network and Multimedia Technology (IC-BNMT), 2011 4th IEEE International Conference on* (pp. 80-83). IEEE.

[8] Younglove, R.W., 2001. IP security: what makes it work?. *Computing & Control Engineering Journal*, *12*(1), pp.44-46.

[9] Liyanage, M., Oulu Finland 2016."Enhancing security and scalability of Virtual Private LAN Services", Available online: http://jultika.oulu.fi/files/isbn9789526213767.pdf

[10] Dhall, H., Dhall, D., Batra, S. and Rani, P., 2012, January. Implementation of IPSec protocol. In *Advanced Computing & Communication Technologies (ACCT), 2012 Second International Conference on* (pp. 176-181). IEEE.

[11] Kent S (2005). IP Encapsulating Security Payload (ESP). RFC 4303

[12] Narayan, S., Williams, C.J., Hart, D.K. and Qualtrough, M.W., 2015, January. Network performance comparison of VPN protocols on wired and wireless networks. In *Computer Communication and Informatics (ICCCI), 2015 International Conference on* (pp. 1-7). IEEE.

[13] Wouters, P. and Bantoft, K., 2006. Openswan: Building and Integrating Virtual Private Networks.

[14] Hu, Mingming, et al. "Research and implementation of layer two tunneling protocol (L2TP) on carrier network." *Broadband Network and Multimedia Technology (IC-BNMT), 2011 4th IEEE International Conference on*. IEEE, 2011

[15] Niu, Yan, Jun Li, and Lala Li. "Research on authentication security of wireless local area network based on L2TP protocol." *Services Science, Management and Engineering, 2009. SSME'09. IITA International Conference on*. IEEE, 2009

[16] Jing, Shan, et al. "Study on VPN Solution Based on Multi-campus Network." *Information Technology in Medicine and Education (ITME), 2016 8th International Conference on*. IEEE, 2016

[17] A. Jain, S. N S, S. K. Lohani and M. Vutukuru, “A Comparison of SDN and NFV for Re-designing the LTE Packet Core,” 2016.

8. APPENDICES

Appendix 1. ipsec.conf file configuration

```
config setup
   cachecrls=yes
   uniqueids=yes
   charondebug=""

conn %default
   keyingtries=%forever
   dpddelay=30s
   dpdtimeout=120s

conn L2TP
   left=193.166.28.151
   leftprotoport=17/%any
   rightprotoport=17/%any
   right=%any
   authby=secret
   ikelifetime=1h
   keylife=8h
   ike=aes128-sha1-modp1536,aes128-sha1-modp1024,aes128-md5-
modp1536,aes128-md5-modp1024,3des-sha1-modp1536,3des-sha1-modp1024,3des-
md5-modp1536,3des-md5-modp1024
   esp=aes128-sha1-modp1536,aes128-sha1-modp1024,aes128-md5-
modp1536,aes128-md5-modp1024,3des-sha1-modp1536,3des-sha1-modp1024,3des-
md5-modp1536,3des-md5-modp1024
   auto=add
   keyexchange=ike
   type=tunnel
```

Appendix 2. ipsec.secrets file configuration

```
# This file holds shared secrets or RSA private keys for authentication.

# RSA private key for this host, authenticating it to any other host
# which knows the public part.  Suitable public keys, for ipsec.conf, DNS,
# or configuration of other implementations, can be extracted conveniently
# with "ipsec showhostkey".

193.166.28.151 %any : PSK "123456"
```

Appendix 3. xl2tpd.conf file configuration

```
[global]
ipsec saref = no
```

```
debug tunnel = no
debug avp = no
debug network = no
debug state = no

[lns default]
ip range = 192.168.4.16-192.168.4.17
local ip = 192.168.4.15
require authentication = yes
name = l2tp
pass peer = yes
ppp debug = no
pppoptfile = /etc/ppp/options.xl2tpd
length bit = yes
unix authentication = yes
```

Appendix 4. l2tp.secrets file configuration

```
# Secrets for authenticating l2tp tunnels
# us    them    secret
# *     *       interop
*      *       123456
```

Appendix 5. options.xl2tpd file configuration

```
ipcp-accept-local
ipcp-accept-remote
ms-dns 4.2.2.1
ms-dns 4.2.2.2
auth
idle 1800
mtu 1200
mru 1200
nodefaultroute
lock
proxyarp
connect-delay 5000
name l2tp
login
refuse-pap
refuse-chap
refuse-mschap
require-mschap-v2
```

Appendix 6. chap-secrets file configuration

```
# Secrets for authentication using CHAP
# client        server  secret                  IP addresses
```

```
#masum    *   "masum04"  *
munim    l2tp   "1234asd"  *
```

Appendix 7. Iptables rules in VPN machine

```
Chain POSTROUTING (policy ACCEPT 55 packets, 5853 bytes)
 pkts bytes target     prot opt in     out     source               destination
   30  2188 SNAT       all  --  *      eth0    192.168.4.16         0.0.0.0/0
to:192.168.254.132
```