vEPC 3.0 over Kernel Bypass
User Manual

The following components, their setup and set of experiments are kernel bypass version of vEPC version 1.1. The LTE EPC components of the previous version are rewritten in event driven programming paradigm to run on kernel bypass for evaluating performance of these components in these setups. The User manual contains steps to create setup, install the components and run experiments. For details on understanding the code and programming logic, please look at the developer manual.pdf under doc folder.

Setup Overview:

The Kernel Bypass EPC is built for Ubuntu/Linux machines. Each component of the EPC is run on separate virtual machines on a server. We measure control plane and data plane performance of these components. The setup for the data and control plane is slightly different. Unless specified, consider the steps as common for both the setups.

The setup consists of Vale Bridge (Netmap based switch) running on the host and netmap module running on each VM. The kernel bypass components i.e. MME, SGW, PGW use mTCP for handling tcp/ip processing. Also for data plane, we use kernel bypass ran simulator and sink for data transfer and so, for data plane experiments, RAN and Sink will also run over mTCP.

Installation:

Follow these steps in each of the six VMs.

Netmap Installation
Refer to following documentation for netmap installation:
https://github.com/networkedsystemsIITB/Modified_mTCP/blob/master/mTCP_over_Netmap/docs/netmap_docs/user_manual.pdf
Install Vale based packet I/O distribution in host and single core netmap installation for guest VMs.

mTCP Installation

Follow these steps in each of the six VMs.

1. Download and install mTCP from github. Use instructions as described in github to install mtcp.

   Link: https://github.com/eunyoung14/mtcp
2. Copy the EPC components in the mtcp/apps folder.

Application Installation:

Follow these steps in each of the six VMs.

1. From the support folder, run install.sh. This will install all the software modules/tools required for proper compilation and working of vEPC

Control Traffic Setup and Experiments:

You will need 5 VMs to run the control plane experiments. Kernel Bypass components (MME, SGW and PGW) are run in single core VMs and remaining (Ran and HSS) can be run as per load requirements. The configuration for which we carried out experiments are as follows

EPC Component	CPU Cores needed	RAM
RAN	4 cores	4 GB
HSS	4 cores	4 GB
MME	1 core	4 GB
SGW	1 core	4 GB
PGW	1 core	4 GB

Follow following steps for setting up components. We assume all installations are done and netmap in guest is connected to vale bridge in host.

Steps common for all EPC components

1. Place the source code folder (EPC_v1.1 control plane) in the mtcp/apps folder of all five VMs.
2. In the defport.h file, change the IP/Port as per the VMs.

Steps specific for MME, SGW and PGW VMs

1. Edit server.conf to change "eth" to be used, and memory buffers.
2. Go to include-epc folder and run "make clean && make".
3. Go to control plane folder and run "make mme" for MME VM, "make sgw" for SGW VM, "make pgw" for PGW Vm.

Steps specific for HSS VM

1. Create MySQL database hss in the VM assigned for HSS.
   ```
   $ CREATE DATABASE hss
   ```
2. Copy the hss.sql file from support folder and run it to populate the necessary tables in MySQL database. "$ mysql −p −u root hss < hss.sql"
3. Go to EPC_control_plane/hss and edit mysql.cpp file to enter mysql user and password.
4. Run "make clean && make".

Steps specific for RAN VM

1. Go to EPC_control_plane/ran.
2. Run "make clean && make".

Running Experiments:

1. Start each component of EPC except RAN in their respective VMs.
   ```
   ./hss.out 50 (start 50 servers)
   ./mme_kby
   ./sgw_kby
   ./pgw_kby
   ```

2. Run ran simulator passing number of ran threads and duration as arguments.
   ```
   ./ransim.out 10 300
   ```

After the experiment is over, the ran simulator will print the number of registrations, throughput and latency before exiting.

Data Traffic Setup and Experiments

You will need 6 VMs to run the data plane experiments. In data traffic experiments, since all except HSS are kernel bypass, we use 5 single core VMs and one VM for hss with any number of cores. The configuration for which we carried out experiments are as follows

EPC Component	CPU Cores needed	RAM
RAN	1 core	16 GB
HSS	4 cores	4 GB
Sink	1 core	16 GB
MME	1 core	4 GB
SGW	1 core	16 GB
PGW	1 core	16 GB

Follow following steps for setting up components. We assume all installations are done and netmap in guest is connected to vale bridge in host.

Steps common for all EPC components

1. Place the source code folder (EPC data plane) in the mtcp/apps folder of all six VMs.
2. In the defport.h file, change the IP/Port as per the VMs.

Packet Forwarding: To allow packets going from Ran to Sink to pass through SGW and PGW, we change MAC headers in the mtcp processing. This is done to emulate packet forwarding behaviour of EPC specification.

Steps specific to RAN, SGW, PGW, Sink

1. In support folder you will find data_ran, data_sgw, data_pgw, data_sink folders. These folders each contain one or two files that will modify original behaviour of mtcp to

forward RAN packets to Sink via SGW and PGW. In each of these files you will find comment and macro below that asking you to put MAC or IP address of particular component. For eg, In “#define DP_RAN_IP” macro, put IP address of Ran simulator VM. In case of “#define DP_SGW_MAC” put MAC address of SGW VM. Also you may find “goto line xxx” in the comment above such macros. Go to that line number in same file and put mac address of that particular component.

2. Copy these files in respective VM's mtcp/mtcp/scr/ folder to replace with original files.
3. Go to mtcp folder and run "sudo make clean && sudo make" to compile mtcp with packet forwarding logic.

Steps specific to RAN, MME, SGW, PGW and Sink VMs

1. In the EPC data plane folder, edit server.conf to change "eth" to be used, and memory buffers etc.
2. In the config/arp.conf modify the arp entries to reflect your VM IP:MAC entries
3. Go to include-epc folder and run "make clean && make"
4. Go to EPC data plane folder and run "make mme" for MME VM, "make sgw" for SGW VM, "make pgw" for PGW Vm, "make ran" for RAN Vm and "make sink" for Sink Vm.

Steps specific to HSS VM

1. Create MySQL database hss in the VM assigned for HSS.
   ```
   $ CREATE DATABASE hss
   ```
2. Copy the hss.sql file from support folder and run it to populate the necessary tables in MySQL database.
   ```
   $ mysql −p −u root hss < hss.sql
   ```
3. Go to hss folder in EPC data plane folder and edit mysql.cpp file to enter mysql user and password.
4. Run "make clean && make".

Running Experiments:

1. In kby_sink.cpp and kby_ran.h change the number of ran threads you want to run, duration of experiments and packet size in respective VMs.

2. Start each component of EPC except RAN in their respective VMs.
   ```
   ./hss.out 5 (start 5 servers)
   ./mme_kby
   ./sgw_kby
   ./pgw_kby
   ./sink_kby
   ```

3. Run ran simulator.
   ```
   ./ran_kby
   ```

After the experiment is over, the ran simulator will exit. In the sink VM, press "ctrl+c", it will print the speed in gbps.