Evaluation of virtualizing DOCSIS MAC by software in data center

Li Zhang, Xin Yang, Lifan Xu
Huawei Technologies Co.,Ltd.

Abstract

Virtualized CCAP was thought as the best cable access architecture in the future, but now the industry hesitates in what parts of the CCAP functionality should be moved to the cloud and what parts should be kept in the physical hardware. It is not only related with technical feasibility of what can be done by software, but also economic consideration.

We all agree on moving the CCAP configuration and subscribe management modules to the cloud, and remaining the DOCSIS PHY in the physical hardware. But as to the DOCSIS MAC, it is not easy to make that kind of decision because the MAC is part of hardware like Encryption and Framer, and part of software like scheduling.

This paper will look deep into each functionality module of the DOCSIS MAC including data plane, control plane and management plane, then give an analysis of virtualized DOCSIS MAC base on the evaluation of generic computing resource spent by each MAC module.

1. DOCSIS MAC and NFV Platform

DOCSIS MAC consists of data plane, control plane, and management plane. Now the control and management plane are all software in CCAP implementation, but the data plane is almost hardware to guarantee high throughput and low latency. Virtualized CCAP is considering about moving the DOCSIS MAC to cloud, before that, we need to think about the cost and performance of virtualized MAC in first.

In order to estimate how many generic CPU cores are needed for DOCSIS MAC virtualization and what performance it can achieve, we break down the DOCSIS MAC to several functionality modules, then build a NFV simulation model driven by packets for each module. We can get the CPU cycles consumption of the whole upstream and downstream MAC data path by simulation, based on that data, we can know the throughput performance in various packet size with a given X86 server.

1.1 DOCSIS MAC Break Down

DOCSIS MAC is a quite complex protocol compared to other access MAC technologies like PON, Figure 1 show the whole DOCSIS MAC modules which includes Upstream and Downstream Data Plane, DOCSIS MAC Control and Management Plane.

Now the DOCSIS downstream and upstream data plane are all hardware implemented by FPGA or ASIC chipset. According to DOCSIS MULPI specification, the downstream frame processing includes the following steps.

1) ***Ethernet RX MAC:*** it receives L2 Ethernet packet from IP router.
2) ***Service Flow Classifier:*** it searches the MAC address table and get destination CM ID firstly, then determine the SFID of the packet base on CM classifier profile, meanwhile, get the QoS attributes.
3) ***Downstream Scheduler:*** it puts packets to different downstream queue according to the QoS attribute, and schedule them out base on specific algorithms and rules.
4) ***Bonding Channel Distributor:*** it distributes packets to different downstream bonding channel base on load balance strategy, the DOCSIS MAC management packets usually have higher priority than data packets.
5) ***AES/DES Encryption:*** it encrypts packet according to the CM individual keyword.

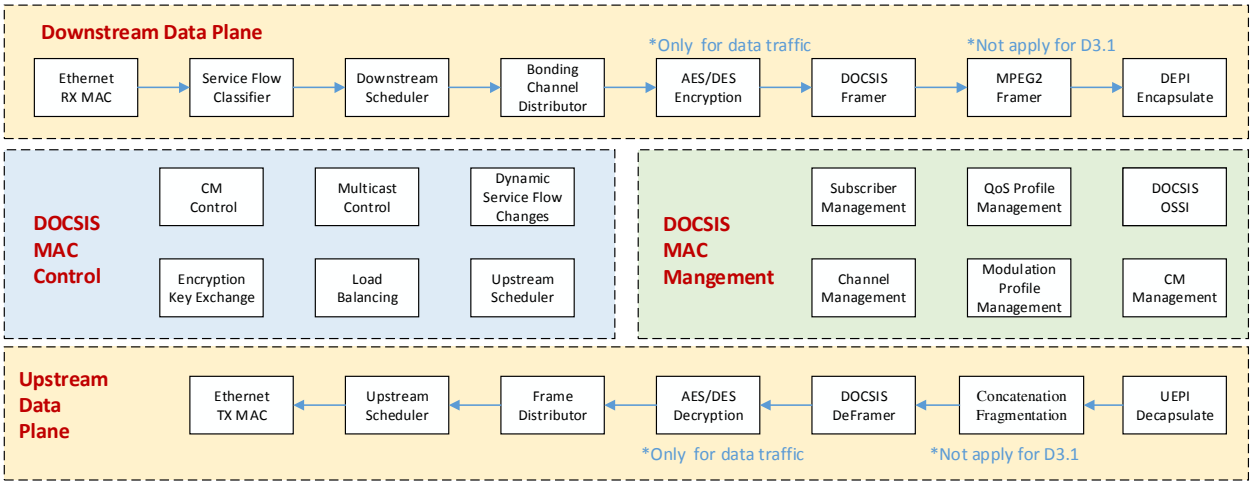

Figure 1. DOCSIS MAC Functionality Break Down

6) ***DOCSIS Framer:*** it adds DOCSIS Header for each packets.
7) ***MPEG2 Framer:*** it slices the DOCSIS stream to 184bytes and adds 4 bytes MPEG Header. It is only for D3.0 and below version because D3.1 has removed the MPEG framer.
8) ***DEPI Encapsule:*** it encapsule the downstream DOCSIS or MPEG frames into L2TP tunnel connection between MAC and PHY.

The upstream data plane is almost reverse path of downstream except some small differences. The processing steps are explained in the below.

1) ***UEPI Decapsule:*** it stripes L2TP header and get DOCSIS or MPEG frames.
2) ***Concatenation and Fragmentation:*** it concatenates the DOCSIS upstream burst segments.
3) ***DOCSIS DeFramer:*** it checks the CRC, then stripes the DOCSIS header.
4) ***AES/DES Decryption:*** it decryptes upstream frames for each service flow.
5) ***Frame distributor:*** it pickes up the DOCSIS management messages from data traffic and forward them to control and management software.
6) ***Upstream Scheduler:*** it schedules the upstream packets into different priority queue buffer.
7) ***Ethernet TX MAC:*** it forwards the upstream L2 packets to IP router.

DOCSIS MAC control plane modules are real-time dynamic operation, unlike static service configuration, they are triggered by service application or MAC internal messages, for example VOIP initialize a call, CM request a upstream bandwidth, or multicast video apps request a program. Those modules are software threads and running on CPU with real-time OS or NP, they may not consume too much CPU cycles but always be required to response immediately. Actually the DOCSIS control functionality is fully defined in MULPI specification, a brief explaination of each control modules is provided in the below.

1) ***Upstream Scheduler:*** it collects Bandwidth Request Message from CMs and calculates upstream grants for each CM in every scheduling cycle.
2) ***Dynamic Service Flow Changes:*** it creates, changes or deletes service flow dynamically, usually the policy server will instruct CCAP to change service flow.

3) ***Load Balancing:*** it is a feature of CCAP that controls the dynamic changes to the set of downstream and upstream channels used by a CM.
4) ***Multicast Control:*** it responses multicast group join, leave request from CPE.
5) ***Encryption Key Exchange:*** it updates CM encryption key words periodically complying to BPI+ specification.
6) ***CM Control:*** it authorize CM when power on and monitor the connection between CM and CCAP.

DOCSIS MAC management plane is about configurating different kinds of profiles like channels, MAC domain service groups, QoS, D3.1 modulation orders and so on. Usually almost configurations are few possibility to be changed unless operators want to extend the upstream or downstream channel number, add QoS profile for a new service etc., so the management modules are all regular software threads which don't have strict real-time requirement.

1.2 NFV DPDK Framework

NFV(Network Function Virtualization) is trying to replace network equipment with generic x86 CPU and software, the hardware can be physical server or virtual compute resource in the cloud. Operators like NFV because it decouples the network functions from proprietary hardware appliances and offer a new way to design, deploy and manage networking services.

DPDK(Data Plane Development Kit) is a widely used NFV programming framework for Intel x86 which enables faster development of high speed data packet networking applications. It provides a set of data plane libraries and Ethernet drivers that run in linux user space for fast packet processing. In order to evaluate virtualized DOCSIS MAC, we change the orginal FPGA MAC data plane modules to software unit base on DPDK framework, and try to get the CPU instruction cycle consumption of each MAC packet processing unit.

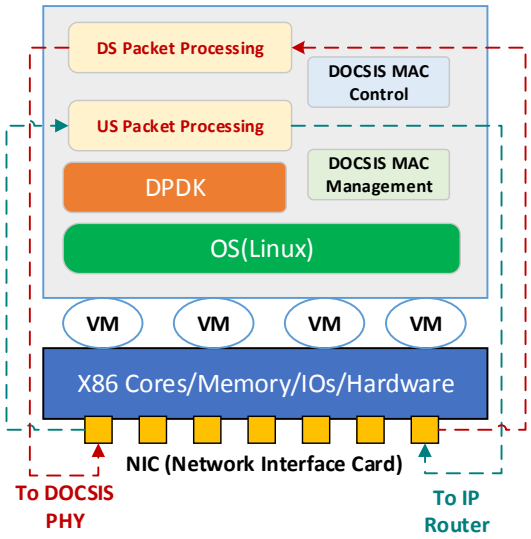

Figure 2. Virtualized DOCSIS MAC

2. Evaluate DOCSIS MAC virtualization

Building a real and complete DOCSIS MAC software and making a end to end system level testing is a tough work. In this paper, we only do real testing in module unit level and sum them together to evaluate the system performance. The hardware we refer to is Huawei FusionServer RH2288 which can provide Intel Xeon 12 cores 2.6GHz-64bits processor, 8*10GE NIC and 128G RAM.

2.1 MAC data plane evaluation

The data plane evaluation is the most difficult, you should design the software processing flows and tasks very well to achieve high throughput and low latency. In

order to make the evaluation as exact as possible, we use the optimized API in DPDK library to implement queue scheduling, classification and packet slicing modules which can guarantee high processing performance, and as to AES encryption and decryption, we choose intel AES-NI library which is a special instruction set designed for security applications.

The MAC data plane modules process and forward traffic by packets, the performance of forwarding software relates with packet size, we choose 64, 128, 256, 512, 1024, 1518 and 2000 bytes as input packet size, then count the processing cycles. Obviously longer packets can get higher bits/s throughput, Table 1 and Table 2 provide the downstream and upstream processing cycles for 2000 bytes frame which is the maximum PDU size in DOCSIS 3.1 MULPI specification. We can see the downstream and upstream computing cost are very close, and AES encryption and decryption consume more than half computing resource.

DS Data Plane Modules	CPU Cycles
Eth RX and Classifier	1948
Downstream Scheduler	1318
Bonding Channel Distributor	218
AES Encryption	16384
DOCSIS Framer	353
MPEG2 Framer slicing	7631
DEPI tunnel	651
Total	28503

Table 1. DS data plane CPU cycles

US Data Plane Modules	CPU Cycles
Eth TX	1552
Upstream Scheduler	818
Frame Distributor	318
AES Decryption	16384
DOCSIS Deframer	2553
Concatenation Fragmentation	5931
UEPI tunnel	502
Total	28058

Table 2. US data plane CPU cycles

Table 3 gives the total downstream and upstream CPU cycles in different packet length.

Packet Size	DS Cycles	US Cycles
64 Bytes	21065	20672
128 Bytes	21103	20768
256 Bytes	21814	21410
512 Bytes	22602	22244
1024 Bytes	24811	24362
1518 Bytes	27009	26453
2000 Bytes	28503	28058

Table 3. CPU cycles for typical packet length

2.2 MAC control and management evaluation

Unlike data plane, MAC control and management threads are driven by events not packets. The events usually occur randomly and can't be measured like packet throughput, fortunely those modules only require real-time operation but don't cost too much computing resource except upstream scheduler module. Thus we decide to port the upstream scheduler code to X86 processor and evaluate the performance by real testing, but as to other modules, it is hard to port them all to X86 processor, so we estimate computing cost by the CPU load usage experience data in current device.

Upstream scheduler operation is cycle base, it collects all the request from cable modem and then allocates grants in every cycle. The scheduling cycle length will impact data traffic upstream latency, usually the scheduling cycle is set to be 1.5-2ms. The computing resource consumption of upstream scheduler relates with how many cable

modem it supports. Figure 3 gives a X86 CPU computing usage curve in different number of Cable Modem, the maximum Cable Modem number one CPU core can support is about 1800, so we can estimate the CPU consumption of upstream scheduler is:

$$\frac{X86\ Cores}{1K\ CMs} = 1/1.8 \approx 0.556$$

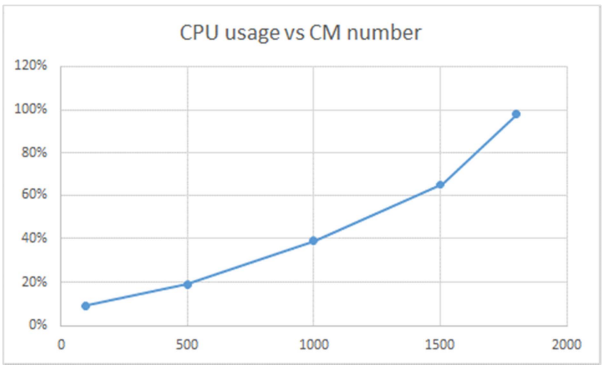

Figure 3. Upstream Scheduler CPU usage

Now we run the other MAC control and management modules except upstream scheduler on a ARM processor with 1.2GHz speed, when we load 1000 CMs and each CM has 4 upstream service flows and 4 downstream service flows, the CPU usage is about 70% in peak time and 20% in idle time. If we think the X86 CPU in data center can be designed to 100% usage, and know the 2.6GHz X86 CPU computing ability is 3.2 times of 1.2GHz ARM through running same testing code, thus we can estimate the X86 CPU cores per 1K CMs for other MAC control and management modules as:

$$\frac{X86\ Cores}{1K\ CMs} = 70\% \div 3.2 \approx 0.218$$

3. Data analysis

Data Plane CPU consumption relates with the packet length, for real data traffic, the Ethernet packet length varies from 64 to 1518 bytes, typically we choose iMIX profile shown in Table 4 as access network packet length distribution model, the CPU consumption can be calculated as:

$$\frac{\text{Cycles}}{bps} = \sum_{l=64-1518} \frac{bits\ percent(l) * Cycles(l)}{packet\ length(l) * 8}$$

Length	64	128	256	512	1518
Packets Percent	50%	10%	15%	10%	15%
Bits Percent	8.9%	3.5%	10.6%	14.1%	62.9%

Table 4. Variable Packet Length Distribution

If we input Table 3 and Table 4 data to the above formula, we can know the upstream data plane is 7.55 cycles/bps and the downstream data plane is 7.7 cycles/bps. Assuming the downstream and upstream bandwidth ratio is 5:1 and the X86 processor speed is 2.6GHz, we can calculates the CPU cores needed to achieve 1Gbps as below.

$$\frac{Cores}{Gbps} = \frac{1G * 7.7 + 1G * 0.2 * 7.55}{2.6G} = 3.54$$

The CPU cores consumption of the whole DOCSIS MAC software can be summed as:

$$\text{Cores} = 3.54 * \text{T} + 0.556 * \text{C} + 0.218 * \text{C}$$

Where:

- T is throughput in Gbps;
- C is the number of cable modem in one thousand.

For the data shown in Figure 4 it gives 3 CPU consumption curve for small HUB with 10K HHPs, typical HUB with 30K HHPs and big HUB with 50K HHP. Given a HUB with 30K subscribers and 80Gbps bandwidth, the virtualized DOCSIS MAC needs about 300 X86 cores which means it needs 25x2U Fusion Server.

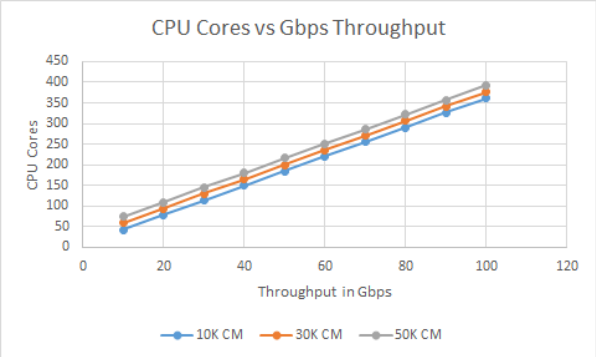

Figure 4. Virtualized MAC CPU consumption

Figure 5 gives the power consumption comparison between the DOCSIS MAC X86 CPU implementation, FPGA implementation and ASIC implementation. We can see the X86 power consumption is 38.35 watts per Gbps bandwidth, it is really a huge number compared to FPGA and ASIC.

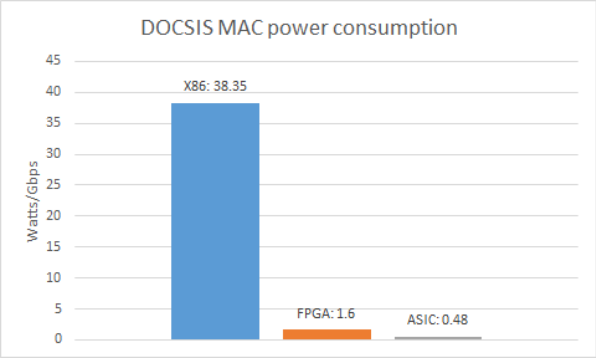

Figure 5. MAC power consumption comparison

4. Summary

NFV changes various network devices to generic hardware and specific software, virtualizing some network devices can get a huge benefit with less pay but some devices may be not.

This paper provides a detailed evaluation of virtualizing DOCSIS MAC by generic CPU, base on the data in this paper, we can see there is still a big gap between DOCSIS CPU implemented MAC and FPGA MAC. The power consumption of X86 CPU MAC is 24 times of FPGA MAC to achieve the same bandwidth, and the hardware cost is about 77 times. We believe the generic hardware will get faster and faster in the future, but 24 times is not the number can be eliminated in short term. Now the generic X86 servers in the cloud are designed for applications which need huge memory and storage, actually the access forwarding device don't require that, that's why the X86 implemented DOCSIS MAC is lack of economic competition. If some new generic hardware platform designed for forwarding device comes out in the future, the DOCSIS MAC NFV dream may come true.